Personal Computer

MZ-2000

OWNER'S MANUAL

SHARP

Personal Computer
MZ-2000

Owner's Manual

ご　注　意

（1）本書の内容は、改良のため変更することがあります。

（2）本書は、誤り、記載もれ等のないよう確認、チェックを行った上で作成しましたが、もしお気付きの点がありましたら、最寄りのシャープのサービス窓口（技術サービス部・サービスステーション・サービスブランチ）までご連絡願います。

（3）本書の内容の一部または全部を当社に無断で転載することは禁じられています。

（4）本書のプログラムを使用したことによる金銭上の損害および逸失利益または第三者からのいかなる請求についても当社はその責任を負いませんのであらかじめご了承ください。

MZ-2000は、シャープの多年にわたるエレクトロニクス技術の経験の粋により完成したコンピュータです。製品は出荷前に充分な検査を受けており規定通りの操作によって直ちに動作しますが、開梱時にはまず輸送中に損傷を受けていないかの点検を行なってください。万一、不都合な点が見つかった場合は直ちに購入された販売店に連絡してください。

また、使用時には、特に次の各注意を守って、本機を末永くご愛用いただきますようお願い申し上げます。

■本機はLSIなど精密な電子部品をたくさん使用しています。極端な高温下や低温下や激しい温度変化のあるところ、直射日光の当たるところ、あるいは振動やほこりの多い場所はさけてご使用願います。

■電源コードを傷つけないようコードを通す場所に注意してください。また電源コードを抜き差しするときは、本体の電源を切り、電源プラグを持って行うようにしてください。

■電源スイッチを瞬時間でON／OFFすると、コンピュータの初期動作が行えないことがあります。スイッチング操作は時間をおいて丁寧に、確実に行ってください。

その他の重要な使用上の注意については、本書巻末A.4にまとめて示されています。

はじめに

この度はシャープパーソナルコンピュータMZ-2000をお買上げいただき誠にありがとうございました。

ご使用になる前に、マニュアルに記載された操作方法、注意事項をよく理解され、正しい操作によって本機をご使用されますようお願いいたします。

パーソナルコンピュータMZ-2000の付属マニュアルは次の2冊があります。

■ Owner's Manual………本書

■ BASIC／MONITOR Manual

本書の第1章および第2章は、パーソナルコンピュータMZ-2000の特徴と一般的な操作方法を解説していますので、はじめにお読みください。続く第3章は、MZ-2000のハードウェア拡張の方法についてまとめています。第4章では、MZ-2000内部のハードウェア構成を回路図と共に解説しており、ユーザの方々が自由なハードウェア拡張、ソフトウェア拡張を行なう場合の参考データとしています。

パーソナルコンピュータMZ-2000のシステムソフトウェアは、全てファイル（カセットテープファイルまたはフロッピーディスケットファイル）によって供給されます。このコンピュータには、標準プログラミング言語BASICを使用するための、BASICインタープリタMZ-1Z001およびMONITOR MZ-1Z001Mが入っている1本のカセットテープが付属しており、その拡張された高度なBASICインタープリタによってMZ-2000の特徴を活かした、多様で自由なプログラミングを行うことができます。その他PASCAL言語や、システム拡張によってDISK BASIC、FDOS等のサポートシステムソフトウェアについても自由なソフトウェア対応ができ、より豊かなプログラミングの世界を拡げることができます。

BASIC言語の文法と、コマンド、ステートメント、ファンクションの各機能およびBASICインタープリタMZ-1Z001をカセットテープファイルからロードしたとき同時にロードされるMONITORプログラムMZ-1Z001Mの機能については、BASIC／MONITOR Mannalを参照してください。

このマニュアルおよび他の1冊のマニュアルは「保証書」「サービス窓口一覧表」と共に必ず保存するようにしてください。

目　次

パーソナルコンピュータ MZ-2000の世界

Chapter 1

コンピュータに何を実行させるか？どのように利用するのか？それは実際殆んど無限の可能性を秘めているにちがいありません。事実、現代社会の中のあらゆる場所でそれぞれの用途のためにコンピュータが使われています。

BASIC、PASCAL、FORTRAN、COBOLといった高級言語による複雑な科学技術計算、あらゆる事務処理、シミュレーション、統計処理が行なわれているし、各種のプラント、ネットワークでは、計測システム、自動制御システムを働かせています。またプログラム開発の現場では、オペレーティングシステム等によって、コンピュータのソフトウェアの世界そのものの開拓と研究が、コンピュータ自身を使って続けられています。そしてこの世界に、LSI技術の高度な発展によって誕生したマイクロプロセッサを用いたパーソナルコンピュータシステムが、急速にその役割を拡大して来ているのです。

あなたはコンピュータに何を実行させるか？という質問に対する答えは非常に様々なものになるはずです。したがってあなたが手にしたこのコンピュータは、まさにそうした広範な利用を可能にするコンピュータなのです。

この章では、パーソナルコンピュータMZ-2000の世界を、その特徴、ハードウェアの拡張、ソフトウェアの展望の順にまとめて解説します。

1.1 MZ-2000の特徴

MZ-2000の特徴をひとことで言うとコンパクトなボディーの中に、すぐれた操作性を実現していることと、様々なソフトウェアへの自由な対応とハードウェアの自由な拡張を徹底して追求していることだと言えます。

CPU (central processing unit) とメインメモリ装置は、コンピュータの中枢に相当しますが、MZ-2000では、最もすぐれたマイクロプロセッサの1つであるZ80A[†)](シャープLH0080A) を使用し、メインメモリは64Kバイト即ちCPUのアドレスバスが直接ランダムアクセス可能な全アドレスを全てRAMで構成しています。従ってメインメモリ空間上には、固定されたプログラムやデータというものがなく、どのようなシステムソフトウェアも外部ファイルからローディングされ、それぞれのプログラムが自由に無駄なくメインメモリを使うことができます。

この中枢部のまわりに、キーボード、カセットテープデッキ等のI/Oデバイス、タイマ、イニシャル・プログラム・ローダなどがすぐれた操作性をもってまとめ上げられています。イニシャル・プログラム・ローダは、コンピュータの電源を入れた時に自動的にスタートし、カセットテープファイルやディスケットファイルからプログラムを読み出し実行します。

コンソール(操作卓)上には、タイプライタ型メインキーボードをはじめとして、数値入力キー、デファイナブルファンクションキー、カーソルコントロールキーおよび前面パネルにカセットテープデッキコントロールキーが配置されキー入力モードによって、ワンタッチで各種のコントロールデータやデータ入力ができます。

ディスプレイ装置は、キャラクタ表示およびグラフィック表示で多彩なデータ表現が可能です。以下の幾つかのパラグラフに於て、これらの特徴をそれぞれ解説します。

図 1-1 パーソナルコンピュータMZ-2000

†) Z80-CPUのクロック周波数の上限を倍の4MHzにしたタイプであり、高速処理が行われます。

1.1.1　メモリ構成の特徴

メインメモリ空間64Kバイトを全てRAM（random access memory）で構成するというやり方は、もともとコンピュータのメモリ構成として最も自然なものといえます。より大型のコンピュータ、たとえばミニコンピュータでは殆どがそうした構成をとっています。即ち、コンピュータに実行させるプログラムは、プログラムテキストも、プログラミング言語もすべてユーザの自由な選択に任されるのです。

実際プログラミング言語には、BASIC言語、PASCAL言語等の各種の高級言語があり、目的によってアルゴリズムの記述に最適な言語を使い分ける必要があるし、機械語によるプログラムを行なう場合は、開発システムプログラムとしてのAssembler、Linker、ディスク装置を用いたオペレーティングシステムをコンピュータで働かせる要求が出てきます。こうしたソフトウェアの各様の使い方をあらかじめ想定した上で、このMZ-2000でも、上位のコンピュータが行っているのと同様のフリーなメインメモリ構成をとっているのです。

それでは、実行させるべきプログラムを選択し、メインメモリ上へ持って来るにはどうすればよいのかというと、あなたは実行すべきプログラムの入っているカセットテープをカセットテープデッキにセットしておく、あるいはフロッピディスク装置を周辺機器に持っているなら、システムソフトウェアのマスターディスケットをセットしておくと、MZ-2000のIPLプログラムが電源ONと同時にスタートして、自動的にプログラムの初期ロードと起動を行ってくれます。この自動操作は、カセットテープを用いた場合平均2～3分で、フロッピーディスケットを用いた場合わずか数秒で終了します。

IPLプログラムは、メインメモリとは別のアドレス空間上のROMにあり、電源ON時にだけCPUをコントロールします。

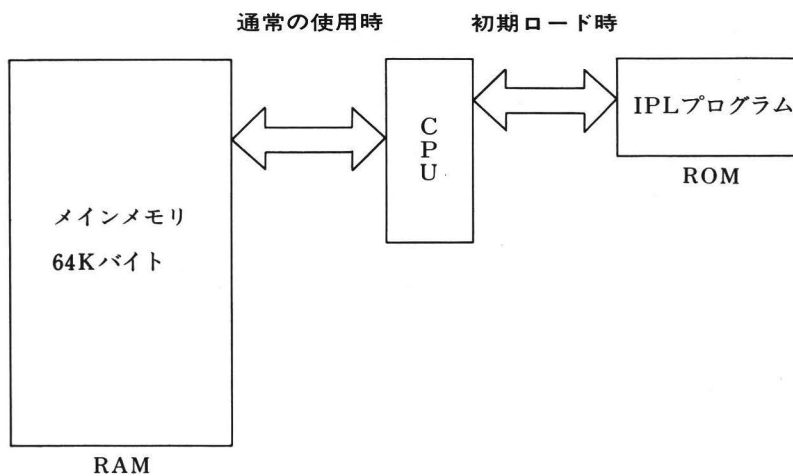



図　1－2　メインメモリとIPLプログラム

1.1.2 MZ-2000のすぐれた操作性と処理能力

パーソナルコンピュータMZ-2000の第1の特徴として、MZ-2000が、1つのプログラミング言語に固定されない汎用コンピュータであることを述べましたが、第2に、どのシステムソフトウェアを使う場合も、すぐれた操作性とハードウェアの処理能力によって効率のよいプログラミングや処理が行えるという特徴があります。

■ 図1－3に、MZ-2000のコンソール（操作卓）を示しています。タイプライタ型メインキーボードを中心にして、その右側に数値入力キー、上部に10個のデファイナブルファンクションキー、左右上下の矢印がついたカーソルコントロールキーがそれぞれグループ分けして配置されています。また、カセットテープコントロールキーは前面パネルに配置されています。

メインキーボードからはモードコントロールキーによるモード切り替えによって、アルファベットの大文字、小文字、大文字の反転文字、数字とその反転文字、カナ文字、シンボル記号、擬似グラフィックパターンの合計225の異った文字を直接キー入力することができます。アルファベット、数字、シンボル記号の並び、およびカナ文字の並びは、それぞれASCII準拠、JIS準拠の配列を採っています。

スペースバーの左側に、[TAB] キーが備えられており、任意の15個のタビュレーションを設定し活用することができます。

数値入力キーは、たくさんの数値を続けてキー入力する場合便利です。[0] ～ [9] のキーの他に、[00]、[+]、[－]、小数点 [・]、それにエントリキー [ENT] が備えられています。

デファイナブルファンクションキーは、文字通り、プログラマが自由にそのファンクションを決めることができます。インストラクションをワンタッチでコンピュータに与えたり、常時使用する特殊な定数、ストリングメッセージ等をそこに定義して置くことができます。各ファンクションキーのすぐ上には、写真のようにファンクションラベルがセットできるようになっています。

カーソルコントロールキーは、スクリーンエディションを行う際に用いられます。4個のキーを独立した場所に置くことによってカーソル操作が扱い易くなっています。

図 1－3 MZ-2000のコンソール

■ カセットテープデッキはソフトウェアでコントロールされ、IPL 操作、プログラムテキストのセーブ／ロード、データファイルのデータ書き込み／読み出し、巻き戻し、早送りをプログラムでコントロールできます。
各システムソフトウェアのファイルサーチにはAPSS 機構 (Automatic Program Search System) があり、早送りによって所望のファイルをす早く見つけ出す働きがあります。
カセットテープファイルによるデータ入出力は、2000ビット／秒と高速で、しかも高い信頼性を持って実行されます。
前面パネルには、巻き戻し、早送り、ストップ、イジェクトをマニュアル操作するための4個のコントロールキーが用意されています。

■ CRT ディスプレイには、キャラクタ表示の他に、高分解能グラフィック表示を行うことができ、多彩なデータ表示を行うことができます。
キャラクタ表示は、1行の桁数を40キャラクタ／80キャラクタの2通りの表示が可能であり、項目数の多い表をディスプレイする場合など80キャラクタ表示が便利です。キャラクタ表示は、スクローリングエリアを任意に決めたり、画面全体の白／黒表示を反転させることもできます。
オプションのグラフィックボードを本体内にセットすることによって320×200ドットまたは640×200ドットの高分解能グラフィック表示が可能です。BASICインタープリタMZ-1Z001をはじめ、サポートシステムソフトウェアにはグラフィックコントロール文（SET文、LINE文、PATTERN文など）が備えられており、関数曲線の表示、デザインへの応用、漢字、仮名表示といった応用が容易にできます。
さらに、カラーディスプレイを接続し、拡張用BASICインタプリタを使用することにより640×200ドットの高分解能カラーグラフィック表示が可能です（COLOR文、CCOLOR文などによりキャラクターの色、グラフィックの色、キャラクタ／グラフィックのプライオリティおよび白黒、カラー両画面へのデータの重ね合わせ等の指定が可能です)。
また、この場合、本体側のCRTディスプレイも640×200ドット／画面となります。

■ オーディオ回路を内蔵しており、音声出力をデータエントリベル、警告音、効果音などに用いることができます。BASIC、PASCAL 等には、3 オクターブの音域内の任意の楽曲を演奏する機能が用意されており、音楽の自動演奏ができます。
また内蔵時計によってデジタル時計の働きをいつでも使うことができます。

1.2　MZ-2000システムの拡張

MZ-2000は、拡張周辺機器としてフロッピーディスクドライブ、プリンタをはじめとして、カラーディスプレイやマークカードリーダなどの豊かなペリフェラルファミリーをシステムに持っています。

フロッピーディスクドライブMZ-80BF は、5.25インチ両面倍密度フロッピーディスケットを媒体としたデュアルドライブユニットであり、ディスクベースのサポートソフトウェア、DISK BASIC、倍精度DISK BASIC、フロッピーDOS、BASICコンパイラ、JIS漢字コード等を走らせながら、高速で、大容量データ処理を実行することができます。プリンタは、文書ファイル、伝票等の出力、プログラムテキストやCRTディスプレイ表示のハードコピーの出力、グラフィックイメージのプリント等の幅広いプリント能力を備えており、この基本システムは、各種のビジネス、プロジェクト分野で要求されるニーズの土台として充分応えられる能力を持っているということができます。

図1-4は、MZ-2000システムのペリフェラルファミリーの展望を示したものです。グラフィックディスプレイ用コントロールカードと拡張I/Oポートは、MZ-2000本体内に設置場所が用意され、その他のペリフェラルは、拡張I/Oポート上にインターフェイスカードをセッティングする形で簡単に接続できます。

拡張I/Oポートには、同時に合計4枚までインターフェイスカードをセットすることができます。

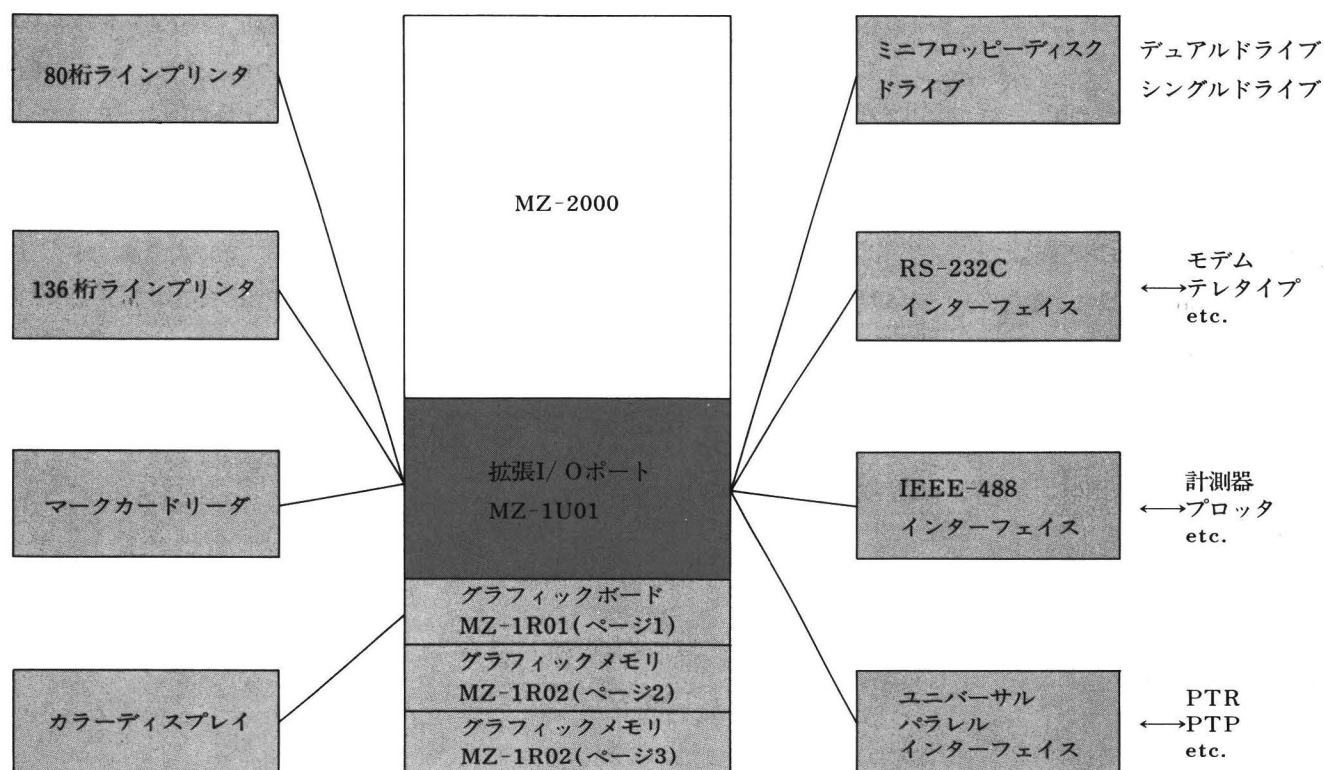



図　1−4　MZ-2000ペリフェラルファミリーの展望

図に示されているように、各種の周辺装置の他に、標準インターフェイスである、RS-232Cシリアルインターフェイス、IEEE-488インターフェイス、8ビットパラレル・ユニバーサルインターフェイスがサポートされ、システムの拡張性を、より柔軟で幅広いものとしています。

MZ-2000の操作方法

Chapter

2

この章は、MZ-2000の操作方法をまとめています。次の順序で解説が行われています。

■ MZ-2000本体の正面図と背面図

■ イニシャルプログラムローディングの方法

■ キーボードの操作

■ ディスプレイコントロール

■MZ-2000の正面図

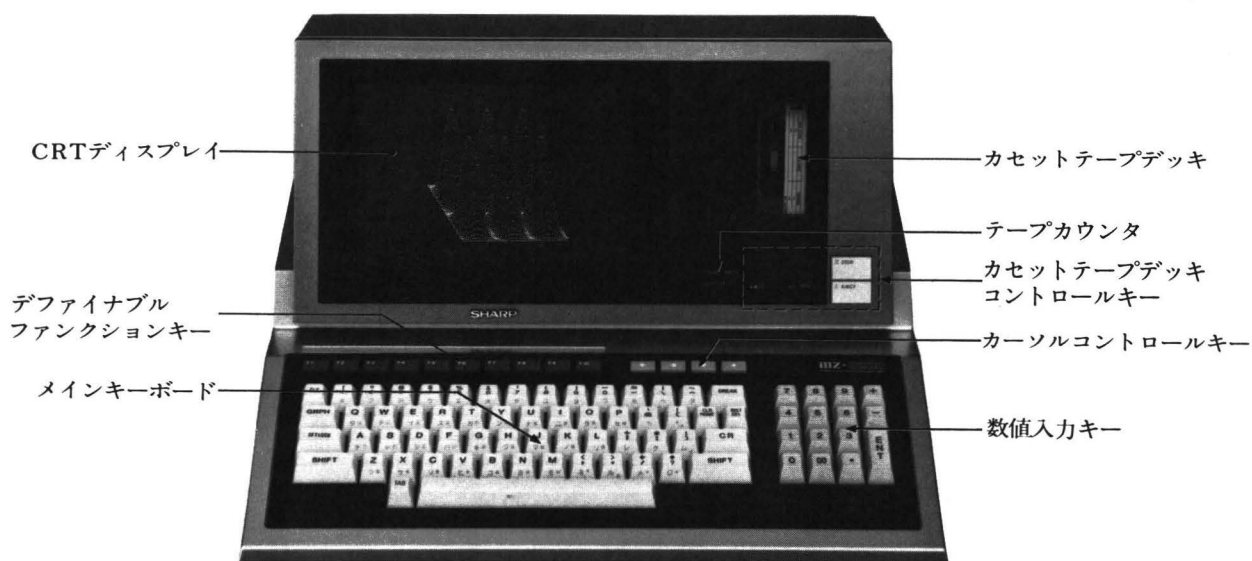



図 2−1

■MZ-2000の背面図

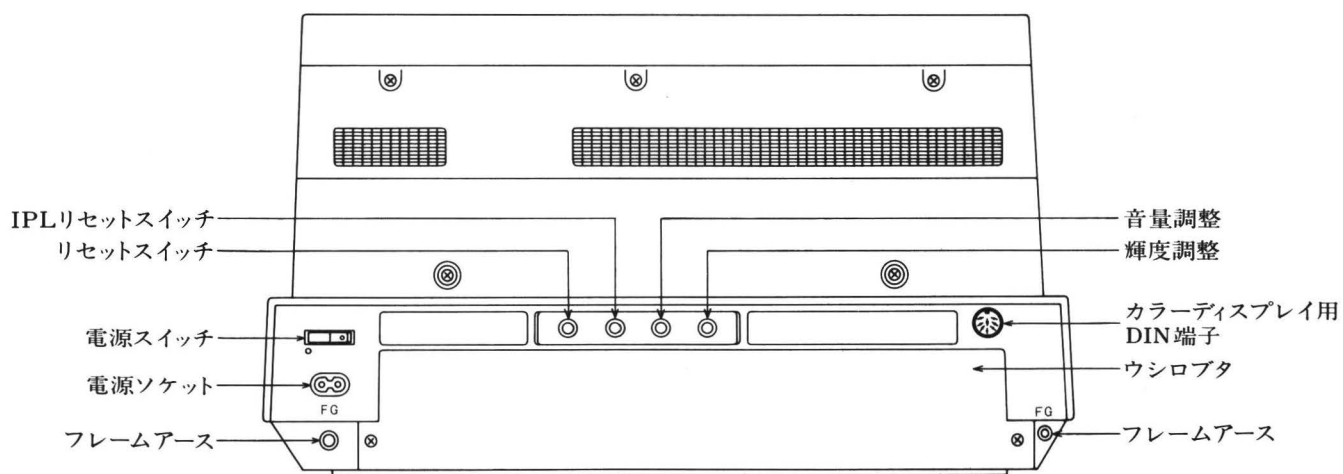



図 2−2

2.1 イニシャルプログラムローディング

MZ-2000はクリーンコンピュータと呼ばれているように、すべてのシステムソフトウェアをファイルによってサポートし、それらの外部ファイルによって各ソフトウェアを起動するシステムを採っています。「MZ-2000の特徴」の項で解説されているように、その際必要となるシステムソフトウェアのローディングは、補助プログラムであるIPL（initial program loader）が実行します。この項では、このIPLの動作と一般的な操作方法について解説を行います。

MZ-2000を使用するには、先ず電源スイッチをONにしますが、IPLはそれと同時に動作を開始し、自動的に目的ファイルのイニシャルプログラムローディングを実行することになります。IPLがカセットテープファイルまたはディスケットファイルからシステムソフトウェアのローディングを終了すると、ロードされたソフトウェアにコントロールが移され、プログラムの起動が完了することになります。従ってプログラマがあらかじめ外部ファイルを所定の入力装置にセットしておけば、MZ-2000の電源をONにするだけで簡単にシステムソフトウェアの起動が行えるようになっているのです。

以下、カセットテープファイル中のシステムソフトウェアの起動と、ディスケットファイル中のシステムソフトウェアの起動について、それぞれの操作方法を解説します。

2.1.1 カセットテープファイル中のシステムソフトウェアの起動

カセットテープファイル中のシステムソフトウェア、たとえばMZ-2000に付属しているBASICインタープリタMZ-1Z001（MONITOR MZ-1Z001Mを同じファイル中に含む）を起動する方法を見てみることにします。

MZ-2000の電源をONにするとIPLプログラムがスタートし、あらかじめカセットテープデッキにテープがセットされていれば、続けて自動的にローディングを開始します。

セットされていない場合は、カセットデッキの蓋が開き、

　　Make ready CMT

とメッセージが表示されます。ここで、BASIC テープをデッキに挿入して蓋を閉めると、続けて自動的にBASICテープのローディングが実行されます。

ローディングをスタートすると、最初にシステムソフトウェアのサーチを行い、次のメッセージを表示します。

　　IPL is looking for a program

BASICをサーチしたら、BASIC（およびMONITOR）をメインメモリへロードし、その時、次のような表示が行われます。

　　IPL is loading BASIC MZ-1Z001

図2－3は、BASICテープのイニシャルローディングを終了して、BASICインタープリタMZ-1Z001が起動した状態を示しています。

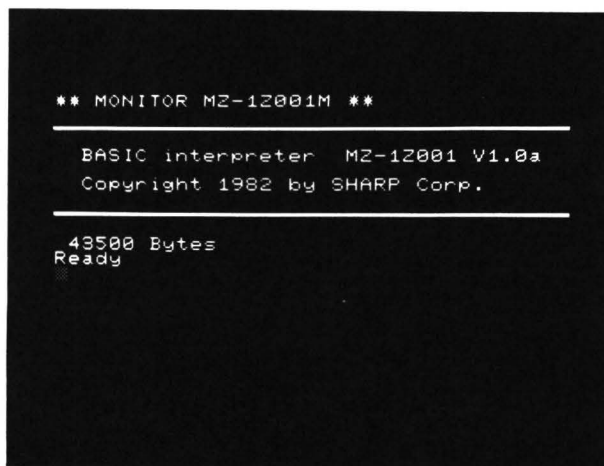



図　2－3　BASICインタープリタMZ-1Z001が起動したことを示すメッセージ

MONITORプログラムのバージョンナンバーは、MZ-1Z001M、BASICインタープリタはMZ-1Z001、BASICテキストエリアのサイズは、43500バイトあること（および、、ソフトウェアの権利がシャープ株式会社にあること）、そしてカーソル点滅によって、システムコントロールがBASICコマンドレベルにあることが示されています。（BASICテープは2分程度でローディングを終了します。）

カセットテープはＩＰＬ動作終了時に自動的に巻き戻されます。

〔注意〕

イニシャルプログラムローディングは、フロッピーディスク装置を接続している場合はディスケットファイル中のシステムソフトウェアの起動が優先となり、その場合、カセットテープ中のシステムソフトウェアの起動は上記の操作とは異る手順が必要となります。２．１．３節の一般の操作手順を参照してください。

2.1.2　ディスケットファイル中のシステムソフトウェアの起動

ディスケットファイル中のシステムソフトウェアを起動するには、勿論フロッピーディスクドライブが正しく接続されていなければなりません。

フロッピーディスクドライブの電源をＯＮにした後システムソフトウェア、DISK　BASIC インタープリタ、倍精度 DISK　BASIC インタープリタ、FDOS等のマスターディスケットを、ドライブナンバ1番のドライブに正しくセットします。その状態でMZ-2000の電源をONにすると、数秒間で各システムソフトウェアが起動されます。

2.1.3 IPL動作のフローチャート

カセットテープファイル、ディスケットファイルのいずれのファイル中にあるシステムソフトウェアも前記の簡単な操作によって、イニシャルプログラムローディングおよび起動が行われますが、この項ではＩＰＬ動作の全体を示すフローチャートに基づいて、特殊なブート操作方法やIPLのメッセージ表示、コントロールの分岐等について詳しく解説を行っています。たとえば、フロッピーディスクドライブを接続していながら、カセットテープ上のシステムソフトウェアを起動する場合や、ディスクドライブ１番以外のドライブによってロードする場合やエラー発生時の処理等について解説されます。

図２－４はＩＰＬ動作の概要をフローチャートに示したものです。前の２項で解説した最もシンプルな自動的なローディング手順は、図の太線に沿った流れによっています。ただし分岐点の条件によってはマニュアル操作が必要なこともあります。

図 2－4 IPL動作フローチャート

フロッピーディスクドライブを接続していながら（即ち、フロッピーディスク用インターフェイスカードを MZ-2000の I/Oポートに挿入している状態）カセットテープからシステムソフトウェアを読み出すには、 [C] キーを押しながらMZ-2000の電源をONにします。（コントロールの流れはαを通過します。）

[C] キーを押さずにMZ-2000の電源をONにすると、ドライブ1番中にマスターディスケットがセットされている場合、それぞれが起動しますが、ドライブ1番が無効状態（たとえば電源が OFFかドライブの蓋があいているかディスケットがセットされていない状態）のとき分岐点βでカセットテープかディスケットかを訊いてきますので、[C] をそこでキー入力することによってもカセットテープからのＩＰＬ動作を行うことができます。

ドライブナンバ1以外のドライブによってＩＰＬ動作を実行させるには、ドライブナンバ1番を無効状態として（ディスケットを入れておかないかフタを開けておくか、或いはドライブの電源を OFFにしておく）ＩＰＬをスタートし、分岐点βに流れを移します。

そこでシステムはあらためて、カセットテープかディスケットかを訊いてきますので [F]（フロッピーディスケット）をキー入力します。続いてイニシャルローディングするドライブ番号を訊いてきますので、該当ナンバをキー入力します。

カセットテープの巻き戻しを行ってからＩＰＬ動作を行う必要のある時は、カセットテープによるＩＰＬ動作をスタートし、続いて [BREAK] キーを押します。[BREAK] キーを押すと、分岐点βへコントロールが移り、自動的に巻き戻しが実行されます。巻き戻しが終ったら [C] をキー入力します。巻き戻しの途中で [C] をキー入力すると、その場所からファイルサーチが行われます。

早送りをする必要のあるときは、カセットテープによるＩＰＬ動作をブレークすると自動的に巻き戻しが行われますので、カセットコントロールの [STOP] キーによってそれを止め、逆に [FF] キーを押して早送りを行います。早送りした後は、[STOP] キーによって止め、[C] をキー入力すると、その場所からファイルサーチが行われます。

ＩＰＬによるシステムソフトウェアの起動が実行された後の動作は、各システムソフトウェアのプログラムに従うので、それぞれのシステムソフトウェアのマニュアルを参照してください。

MONITORやFDOS等の開発プログラムによってユーザが作成したシステムソフトウェアをＩＰＬで起動することも勿論できます。

ＩＰＬはカセットテープファイルをファイル名によってサーチする働きを持っておらず、最初に見つかったシステムソフトウェア（即ちファイルモードがＯＢＪであるもの）をローディングします。たとえばBTX（BASICテキストファイル）等がサーチされた場合は、その場で"File mode error"が発生します。

ＩＰＬ動作時のメモリマップ等は、第4章の技術資料に、またIPLプログラムの全アセンブリリストは別冊のBASIC /MONITOR Manualに示されています。

特殊なイニシャルプログラムローディング方法として、I/O ポート上のメモリボードからのシステムソフトウェアの起動があります。たとえば、BASIC インタープリタ等のシステムプログラムを専用に常時使用する場合、ＩＰＬを瞬時に行うため、それをＲＯＭカード上に置いておき、ＩＰＬによってメインメモリへ高速にブロック転送するという方法です。前記の解説およびフローチャートには示していませんが、スラッシュ"／"をキー入力することにより、この動作がシステムの状態によってスタートします。

2.2　キーボード

この節は MZ-2000のコンソール（操作卓）上の各キーの機能と使い方を解説します。コンソール上には、図2－5に示すように4つのキーのグループがあります。

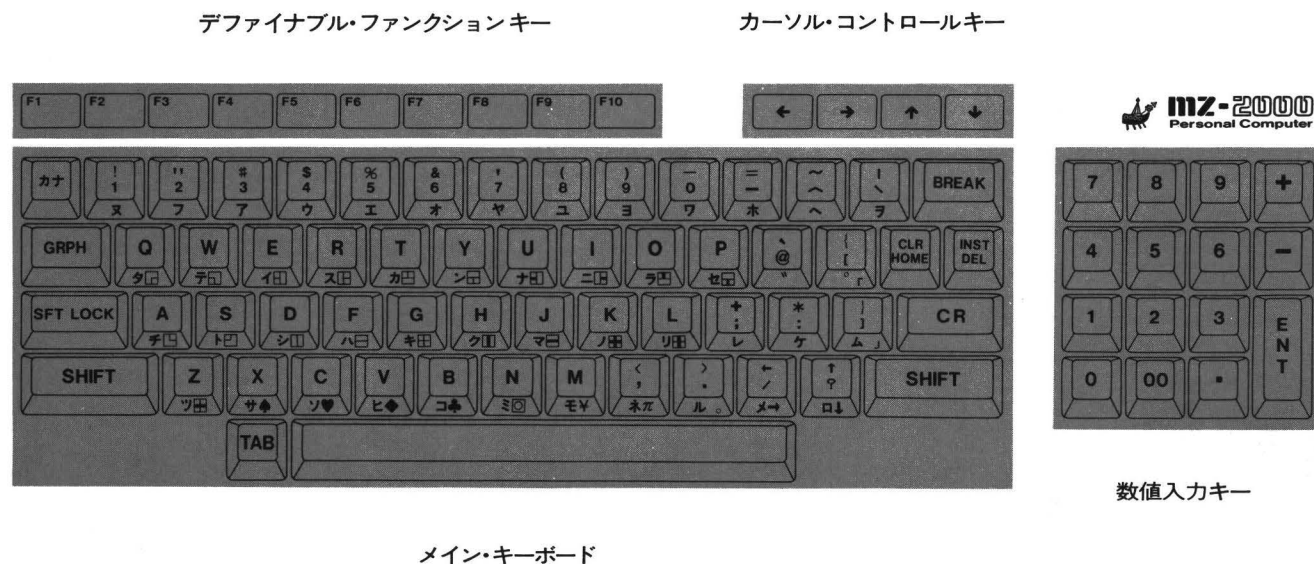



図　2－5　コンソール上のキー配列と4つのグループ

ASCII 準処メインキーボードは、タイプライターのキーボードに似ており、文字の入力用キーと、キャリッジリターン、ブレークなどのコントロール用のキーと、カナ、グラフィックなどのモードコントロール用のキーとからなっています。

数値入力用キーには、数字と＋－記号、小数点、**ENT**（入力）キーが電卓と同じようにまとめられています。

コンソール上部には、左側に青い色の10個のデファイナブル・ファンクションキーがあります。ファンクションキーの働きは、プログラマが任意に決めることができます。

中央の黄色の4個のキーは、カソール移動専用のキーです。

カセットテープデッキの下にある4個のキーは、カセットテープデッキをワンタッチでコントロールするためのキーとなっています。

2.2.1 メインキーボードの使い方

メインキーボードは、幾つかの点を除いて ASCII 準拠タイプライタキーボードに倣ったキー配列をとっています。相違点は、以下の項で説明されるように、キー入力に幾つかの入力モードがあること、それにコントロール用特殊キーが備わっていることです。図2－6はコンソール上のメインキーボードと、そのコントロールキーを示しています。

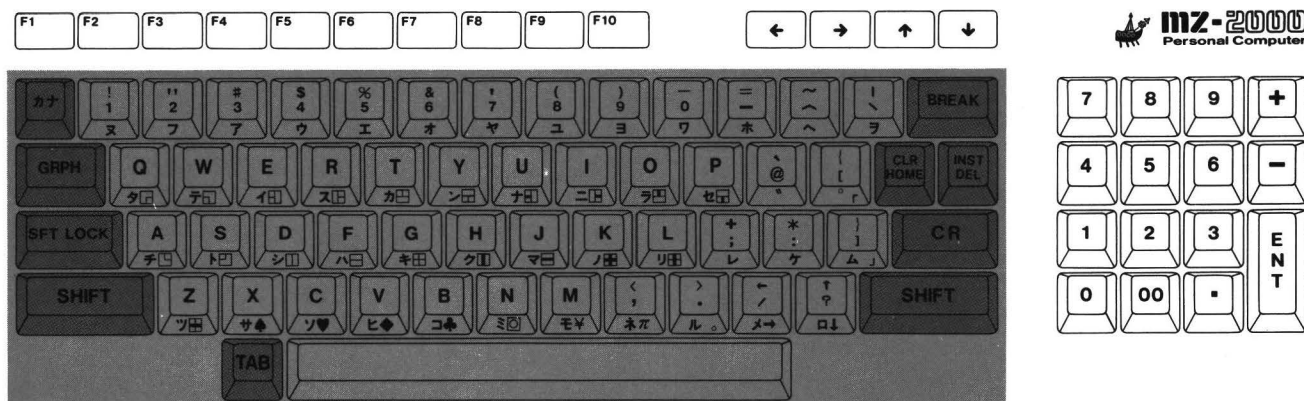



図 2－6 メインキーボードとそのコントロールキー

コントロール用特殊キー以外の通常のキーには、各キーに3個か4個のキャラクタが描かれています。そしてそれらはコントロール用特殊キーによる入力モードの設定状態に従ってそれぞれキー入力されることになります。

キー入力モードには次の3つがあります。

〔1〕 ノーマルモード …………各キーの上面に描かれた ASCII 準拠のキャラクタを入力するモード。アルファベットキーは、大文字／小文字（シフトポジション）が入力されます。

〔2〕 グラフィックモード……… GRPH キーによって設定され、作表のためのパターンやハート、スペードなどのキーの前面右側に描かれているグラフィックパターンを入力します。

〔3〕 カナモード ………………… カナ キーによって設定され、キーの前面左側に描かれている片仮名をキー入力するモードです。

1つのキーを例にとって、キー入力モードが異る時、どのようなキャラクタが入力されるか見ることにしましょう。たとえばZというキーは、図2－7のように全部で6種類のキャラクタの入力ができます。

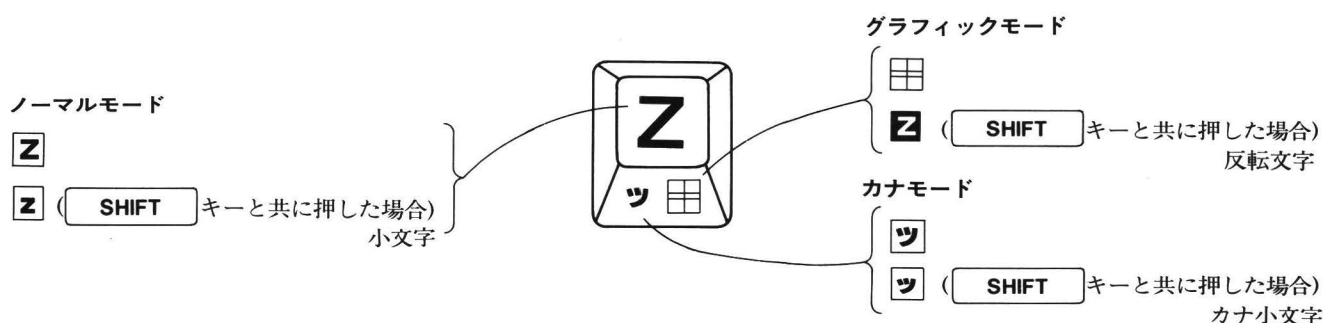



図 2－7 Z のキーによって入力される6つのキャラクタ

次に、コントロール用特殊キーの機能を解説します。

SHIFT

タイプライターのシフトキーと同様に各キーのシフトポジションにあるキャラクタをキー入力する補助キーです。ノーマルモード時には、アルファベットキーについてはその小文字を、その他のキーは上部に描かれているキャラクタを入力します。グラフィックモード時には、アルファベットキーと数字キーについてはその反転文字を入力します。カナモードの場合は、カナ小文字"ァ"、"ョ"などが"ア"、"ヨ"等のキーから入力されます。

BASIC には CHANGE 文が用意されており、この文の実行によってアルファベットキー26文字の大文字／小文字のシフトポジションを逆にすることができます。

CR

キャリッジ・リターンキー。1行のデータ入力を行うキーです。 CR キーを押すとカーソルは次の行の先頭に移ります。数値入力用キーにある ENT キーも CR キーと同じ機能を持っています。(CR：carriage return)

CLR HOME

HOME は、カーソルをCRTディスプレイの左上隅に移します。

CLR は、CRTディスプレイのキャラクタ表示をクリアして、カーソルホームを行います。(CLR：clear)

INST DEL

DELは、カーソルの左隣りの1キャラクタを消し、カーソル位置以降の1行分の文字列を左に1文字分詰める働きがあります。

(DEL：delete)

INST は、カーソル位置にスペース1個を挿入し、カーソル位置以降に続く1行分の文字列を1文字右へ送る働きがあります。

(INST：insert)

BREAK

ブレークコードを入力します。BASIC プログラムの実行中、カセットテープデッキのコントロール中等にこのキーを押すとプログラムやコントロールを中止することができます。

SFT LOCK

シフトポジションをロックするコントロールキーです。このキーを押すとシフトポジションがロックされ、もう一度このキーを押すか CR キー、他のモード切替えキーが押されてキャンセルされるまでシフトポジションのキャラクタがキー入力されます。

(SFT LOCK：shift lock)

GRPH

メインキーボードをグラフィックモードにするモード切替えキーです。このキーを押すとグラフィックモードになります。グラフィックモードでは、各キーの前面右側に描いているグラフィックパターンがキー入力されます。このモードで SHIFT キーを押しながら英数字キーを押すと、英数字の反転文字が入力されます。もう一度このキーを押すか CR キー、他のモード切り替えキーの入力によって解除されます。

(GRPH：graphic)

カナ

メインキーボードをカナモードにするモード切り替えキーです。このキーを押すとカナモードになります。カナモードでは、各キーの前面左側に描いているカナ文字が入力されます。このモードで SHIFT キーを押しながら、"ツ""ヤ""ア"などのキーを押すと、カナ小文字"ッ"、"ャ"、"ァ"などが入力されます。もう一度このキーを押すか CR キー、他のモード切り替えキーの入力によってカナモードはキャンセルされます。

なお、CRTディスプレイのカーソルは、ノーマルモードではチェッカ▩、その他のモードではマル●が点滅します。

TAB

タビュレーションコントロールキー。キー入力の際のタビュレーション操作を行うためのキーです。タビュレーションの設定は、各モニタプログラムのタビュレーションデータエリアを決めることによって行います。

〔1〕 ノーマルモード

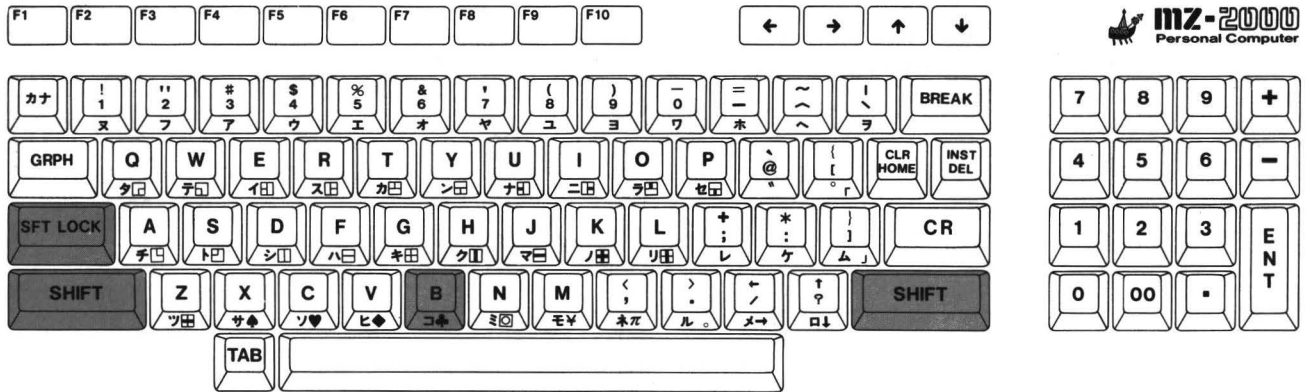



図 2−8

ノーマルモードは、BASIC インタープリタやその他のシステムソフトウェアを起動した時のメインキーボードの最初の状態であり、SHIFT キーを押したシフトポジションと共にアルファベット大文字／小文字、数字、クォーテーションマーク、ドルマーク"＄"、アステリスク"＊"等のASCII標準キャラクタを入力するキーモードです。

たとえばアルファベット大文字の"Ｂ"（図２−８を参照）をキー入力するには B キーを押します。普通のタイプライタでは、大文字をタイプする場合 SHIFT キーを使用しますが、コンピュータプログラミングでは、各種コマンド、ステートメントがアルファベット大文字で表記するので、小文字の方をシフトポジションに置いているのです。(但し前記の BASIC インタープリタの大文字／小文字を逆にする CHANGE 文がそうであるように、ソフトウェアコントロールによって逆にさせることもできます。)

〔2〕 グラフィックモード

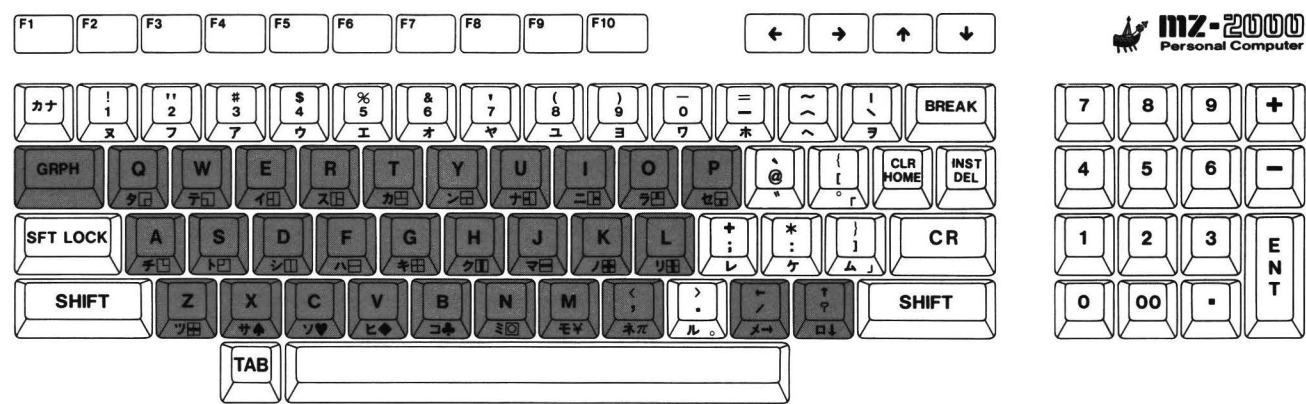



図 2−9 グラフィックキャラクタを持つキー

グラフィックモード、即ち GRPH キーを押した状態のキー入力モードでは、図2−9でアミを施した29個のキーについては、それぞれが持つグラフィックパターン（擬似グラフィックパターン）を入力するためのものです。

たとえば、このモードで、 B キーを押すと、グラフィックパターン"♣"がキー入力されるし、BASICインタープリタのシステム定数"π"(円周率)をキー入力するには、メインキーボードをグラフィックモードにして 〈; キーを押せばよい、ということになります。

図2−10は、グラフィックパターンを用いて表をCRTディスプレイ上に表示させた例です。

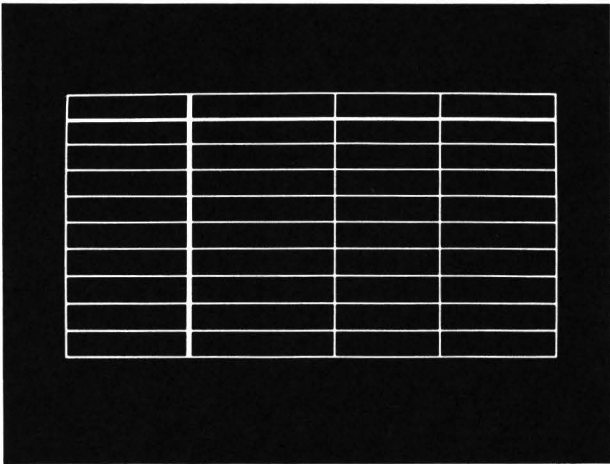



図 2-10 グラフィックモードの表の作成例

グラフィックモードで、 SHIFT キーを使用する場合は、図２－11に示す36個のキー、即ち英字の大文字と数字のキーについてそれぞれの反転文字をキー入力することができます。

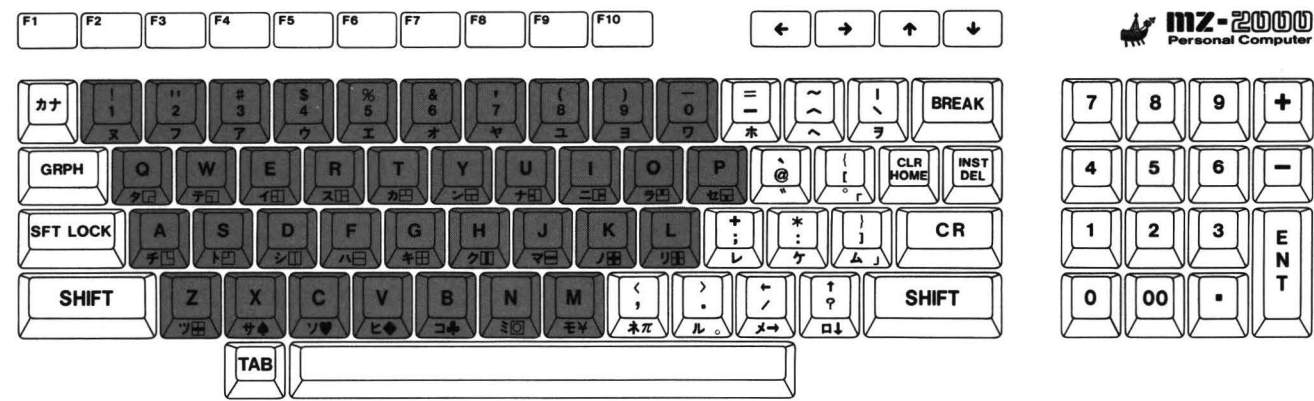



図 2－11 反転キャラクタを持つキー

たとえば、 B キーについては、 SHIFT キーを押しながら押すと、" B " 即ち " B " の反転文字が入力されます。反転文字は、表のタイトル等によく使われます。

図２－12は、図２－10で作った表の枠に反転文字を使って各項目を記入した例です。

Goods in stock

	Unit Price	Piece	Amount

10 April 1981

図 2－12 反転文字を使って項目を表示した例

〔3〕 カナ入力モード

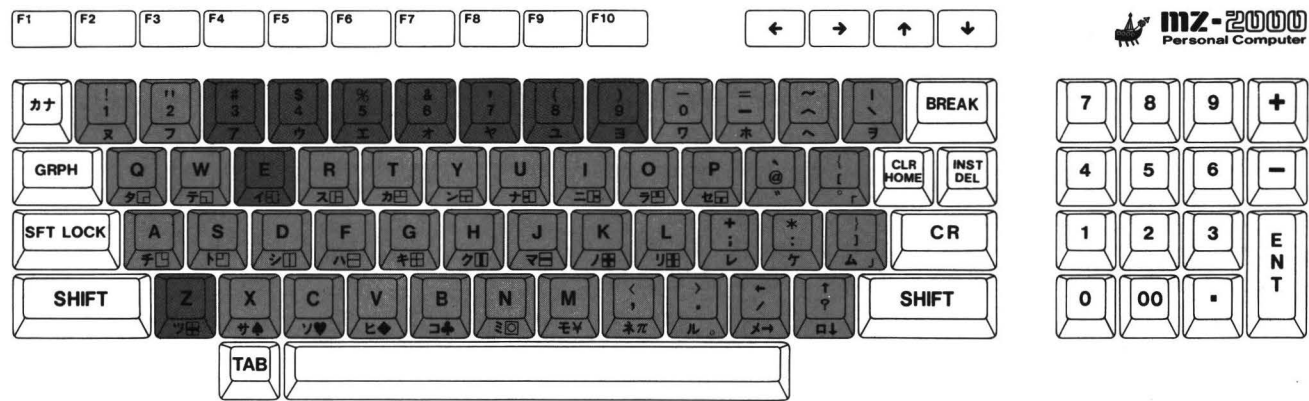



図 2−13 カナ文字の入力キーとカナ小文字の入力キー(濃いアミのキー)

カナ入力は、 カナ キーを押すことによって設定され、図2−13でアミを施した48個のキー (即ちコントロールキー以外の全部のキー) について、それぞれが持っているカナ文字を入力するためのモードです。それぞれのカナ文字は、キーの前面左側に白で示されています。ただし CLR HOME キーの左の2つのキーは、濁点と半濁点を示しています。

このモードでキーを押すと、アミを施したキーについては、それぞれが持っているカナ文字が入力されます。たとえば Z キーは、カナ文字"ツ"を持っているので、このキーを押すと、"ツ"が入力されます。

SHIFT キーを用いるとカナ小文字"ッ"が入力されます。即ち"ツ"の他"ア"、"イ"、"ウ"、"エ"、"オ"、"ヤ"、"ユ"、"ヨ"の各カナ文字は、 SHIFT キーを押しているとそれぞれカナ小文字が入力されます。

図2−14はカナ文字によって表の項目を記入した例です。

ヒョウ.33　ザ゛イコ ヒンモク

	タンカ	スウリョウ	ソウガ゛ク
エンピ゜ツ			
ヘ゜ン			
ノート			
ケシゴ゛ム			
インク			
フウトウ			
ヨウシ			
カッター			
モノサシ			
			1981ネン 4ガ゛ツ 10カ

図 2−14 カナ文字を使って項目を表示した例

2.2.2 数値入力

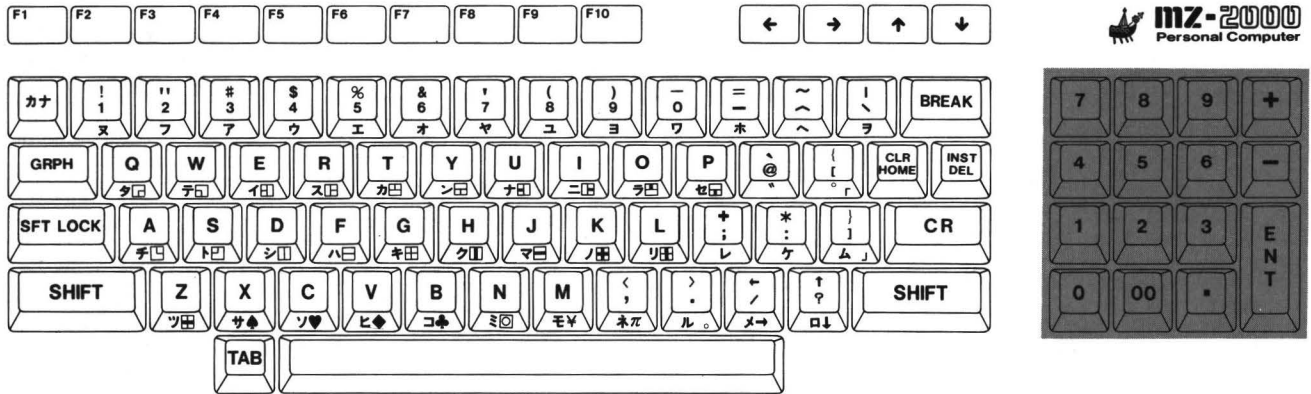



図 2−15 数値入力キー

メインキーボードの右側にまとめられたキーグループを数値入力キーと呼びます。数値入力キーには、10個の数字に加えて0を2つ並べた 00 キー、それに ・ + − ENT キーが置かれています。

・ + − キーはメインキーボード上にもあるし、 ENT キーすなわちエントリーキーは、メインキーボード上の CR キーと全く同じ機能を果たします。[†] にもかかわらずこれらのキーを数値入力キーとしてまとめた（グループとした）のは、たくさんの数値データを入力する場合の便宜を図っているからです。[†] 実際、多くの数値処理応用プログラムでそうした要求が頻繁に生じるにちがいありません。

00 キーを1回押すと、0 キーを2回押した時のように00が入力されます。

5 キーのキートップの中央には、小さな突起がついています。数値入力キーから目を離して操作する時の案内役になります。

数値入力キー上のキーは、メインキーボードのキー入力モードや SHIFT キー操作とは関係がありません。

---

†）ただしキースキャニングにおけるストローブ信号は異ります。従って、機械語プログラムで、数値入力キー上のキーとメインキーボード上の同じキーを区別することは可能です。（ただし CR と ENT は区別できません）。表4−5を参照のこと。

2.2.3　デファイナブルファンクションキー

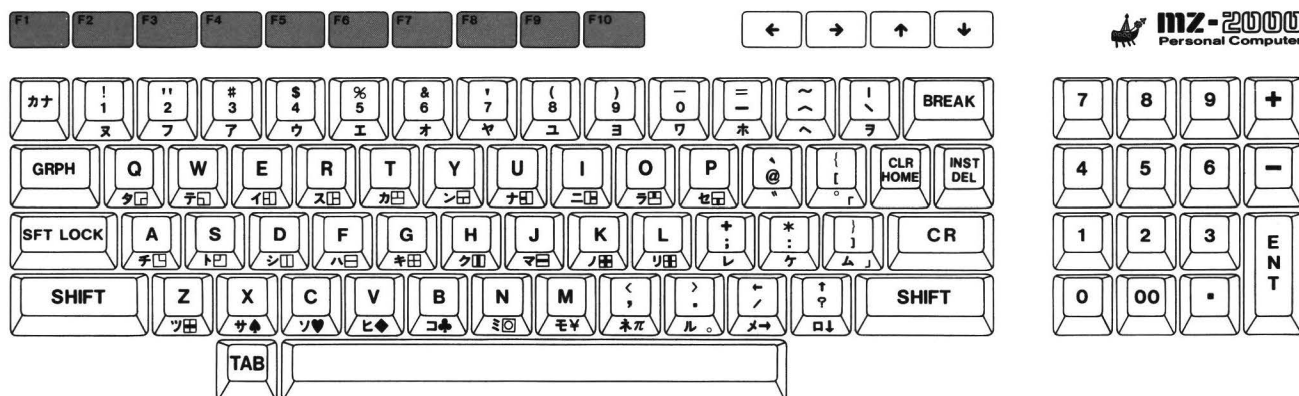



図　2−16

メインキーボードの上に並んでいるＦ１からＦ10までの名前のついた青い色の10個のキーを、デファイナブルファンクションキーと呼びます。

これらのキーは、各システムソフトウェアの起動時に定義されますが、自由にこれらのキーの機能を再定義して使うことができます。たとえばBASIC MZ-1Z001では、DEF KEYステートメントによってデファイナブルファンクションキーの定義を行います。BASICコマンドRUNを、ファンクションキー1番に定義するには次のステートメントを実行します。

DEF　KEY(１)＝RUN

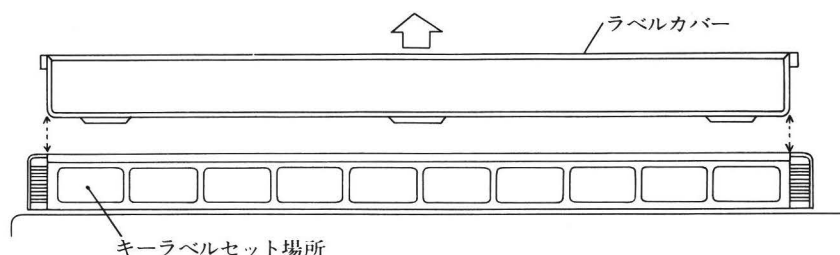
１度このステートメントを実行すると、新たに定義し直さない限りデファイナブルファンクションキー１番は、RUNコマンドと同じ機能を持つことになります。即ち、直接モードでファンクションキー１番を押すと"RUN"の表示が現われます。続いて CR キーを押すとプログラムが実行されます。更に、RUNコマンドプラス CR キーをまとめて定義することができます。それには次のステートメントを実行します。

DEF　KEY(１)＝RUN ⏎

ここで"⏎"という記号は CR キーを押すのと同じ機能を表し SFT LOCK キーと GRPH キーを同時に押すことによって入力することができます。こうして定義しておけばファンクションキー１番を１回押すだけでプログラムが実行されます。ファンクションキーの定義内容はBASICではKLISTコマンドによって調べることができます。デファイナブルファンクションキーには、RUNコマンドのような直接実行命令を定義しておくと便利ですが、その他の任意の数字データやストリングデータを自由に定義することができます。

■ファンクションキーラベルのセット

付属のキーラベルをファンクションキーの後方にあるラベル格納場所にセットしておくと便利です。

透明のラベルカバーを後方へ引き出し各ファンクションキーに対応するラベルをセットしてください。

2.2.4　カーソル・コントロールキー

デファイナブル・ファンクションキーの右側の、矢印記号のある4個の黄色のキーはCRTディスプレイ上のカーソルを移動させるコントロールキーです。

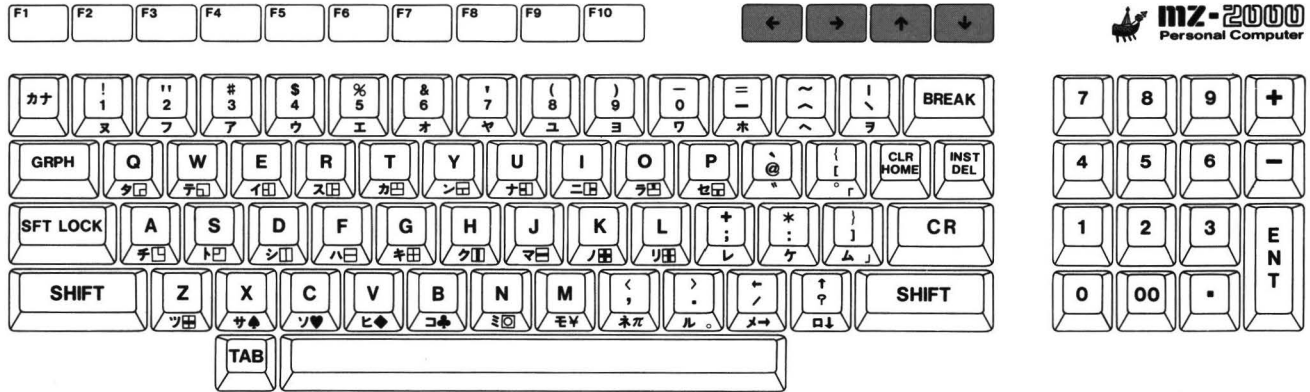



図　2－17　カーソル・コントロールキー

左、右、上、下の各方向へそれぞれカーソルを移動させます。各システムプログラムのスクリーンエディションの際に用います。

BASICインタープリタでは、メインキーボードの SHIFT キーを押しながらカーソル・コントロールキーを押すとオート・リピートが行われスクリーンの離れた所へ早くカーソルを移動させるのに便利です。(システムソフトウェアによっては SHIFT キーを押さなくともカーソル・コントロールキーを押し続ける間オート・リピートするものもあります。)

2.2.5 カセットテープデッキ・コントロールキー

カセットテープデッキの下にある4個のキーはカセットテープデッキのコントロールキーです。

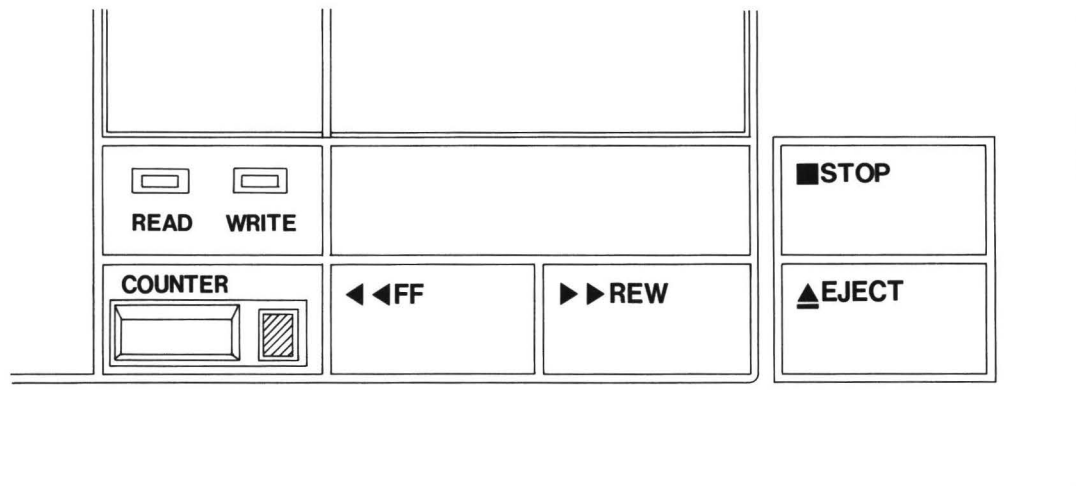



図2—18　カセットテープデッキ・コントロールキー

これらのキーは、カセットテープデッキに直接つながったコントロールキーであり、他のキーとはメカニズムが異っています。†)

キー	機能
▶▶REW	…………カセットテープの巻き戻しを行う（LEDランプが点灯します）
◀◀FF	…………カセットテープの早送りを行う（LEDランプが点灯します）
■STOP	…………停止する
▲EJECT	…………カセットの出し入れのためのイジェクトを行う

これらのコントロールはコンピュータの動作モードとは関係なく実行できます。

読み出し／書き込み時に対応するLEDランプ（READ…緑色、WRITE…赤色）が点灯します。
テープカウンタの右横のボタンはカウンタリセット用です。

カセットテープへのデータの書き込みや、データの読み出しはソフトウェアによるオートマティックコントロールになっています。

BASICインタープリタには、プログラムテキストの書き込み／読み出しはSAVE、LOADコマンド、データファイルの書き込み／読み出しはPRINT/T、INPUT/Tステートメントによって自動的に行われます。また、各システムソフトウェアには必ず書き込み／読み出しのインストラクションが備わっています。

†) 即ち、他のキーは、PIOによるキースキャニング（4.3.3節を参照のこと）によってキー入力が検知されソフトウェアによって処理が行われますが、カセットテープデッキ・コントロールキーは、カセットデッキのモータ制御、イジェクト機構の制御をダイレクトに行います。

2.3 CRTディスプレイコントロール

MZ-2000には、3つのディスプレイコントロールシステムがあります。キャラクタディスプレイコントロールとグラフィックディスプレイコントロールさらにカラーグラフィックディスプレイコントロールです。キャラクタディスプレイコントロールでは、アルファベット大文字、小文字、反転文字、数字、カナ文字、擬似グラフィックパターン等をキャラクタジェネレータを用いて CRT ディスプレイに表示させます。グラフィックディスプレイコントロールでは、オプションのグラフィックボードを用いて、320×200ドット／画面または640×200ドット／画面で任意のグラフィック表示ができます。上記のグラフィックボードに、グラフィックメモリページ2、3を装備後カラーディスプレイを接続し、拡張用BASICを使用することにより、カラーグラフィック表示ができます。

2.3.1 キャラクタディスプレイコントロール

キャラクタ表示は、キャラクタコード（ASCIIコード）によってキャラクタフォントをキャラクタジェネレータから呼び出してCRTディスプレイ上へ表示されます。図2－20には、MZ-2000が持っているすべてのキャラクタのフォントをASCIIコードに対応させて示しています。これらのキャラクタは、前節で述べたように各モードによってメインキーボードから直接キー入力することができます。(ただしチェッカ▩、ベタ■およびマル●を除きます。)

CRTディスプレイのキャラクタ表示には、80キャラクタ表示モードと40キャラクタ表示モードとがあります。各モードでのキャラクタフォント構成は同じですが、80キャラクタ表示モードでは、1文字の大きさが左右½に縮少されます。また画面全体の白／黒反転機構があり、1つのキャラクタ、たとえば "A" は、図2－19に示すような4種類の表示が行れます。

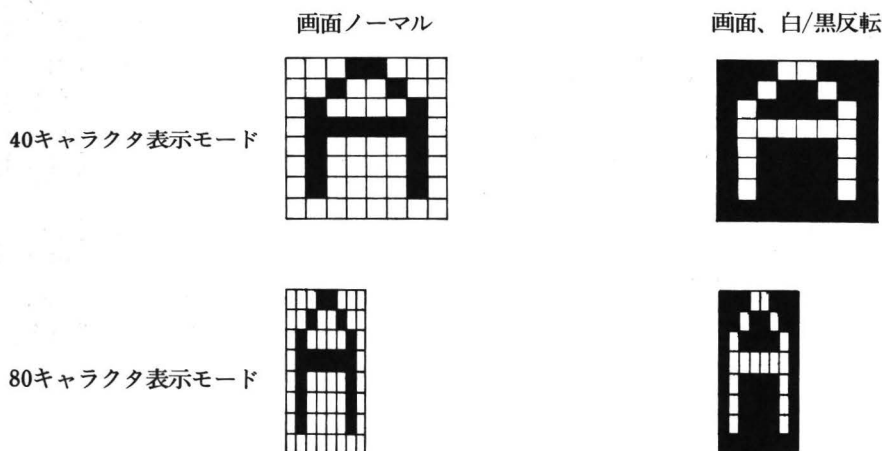



図 2－19 キャラクタ "A" の4種類の表示例

L＼U	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0				0	@	P	`	p	‖	─			タ	ミ		
1			!	1	A	Q	a	q	↓	＊	。	ア	チ	ム		
2			"	2	B	R	b	r	↑		「	イ	ツ	メ		
3			#	3	C	S	c	s	→	●	」	ウ	テ	モ		
4			$	4	D	T	d	t	←	○		エ	ト	ヤ		
5			%	5	E	U	e	u	♠	┐		オ	ナ	ユ		
6			&	6	F	V	f	v	♥	┘	ヲ	カ	ニ	ヨ		
7			'	7	G	W	g	w	◆	┌	ァ	キ	ヌ	ラ		
8			(	8	H	X	h	x	♣	└	ィ	ク	ネ	リ		
9			)	9	I	Y	i	y	├	┼	ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z	┤	│	ェ	コ	ハ	レ		
B			+	;	K	[	k	{	┬	─	ォ	サ	ヒ	ロ		
C			,	<	L	\	l			┴	┴	ャ	シ	フ	ワ		
D			-	=	M	]	m	}	┼	┬	ュ	ス	ヘ	ン		
E			.	>	N	^	n	~	┼	┤	ョ	セ	ホ	゛		
F			/	?	O	￣	o		┼	├	ッ	ソ	マ	゜		

図 2—20 キャラクタフォントによるASCIIコード表

U………上位4ビット

L………下位4ビット

図　2—21　ASCIIコード表：キャラクタおよびコントロールコード

2.3.2 グラフィックディスプレイコントロール

図2－22は、グラフィックディスプレイコントロールをBASICインタープリタによって行った例を示しています。左は、3次元曲線をSET文によって描いたもの、右は、ドイツ文字をPATTERN文によって表示させたものです。BASIC によるコントロールの詳細は、BASIC／MONITOR Manualを参照してください。

図 2－22

2.3.3 カラーグラフィックディスプレイコントロール

キャラクタ、グラフィックデータ、およびバックグランド（背景）に対して、それぞれ表2－1のごとく、計8色の中から自由に色の指定ができます。

*出力データ	指定色
0（8）	黒
1（9）	青
2（10）	赤
3（11）	紫
4（12）	緑
5（13）	水色
6（14）	黄
7（15）	白

*ポート＄F5, ＄F4への出力データ

表 2－1

■ キャラクタのカラー表示

カラーディスプレイ画面全体のキャラクタに対して色を指定できます。ただし、キャラクタ単位での色指定は行えません。

プライオリティ（キャラクタとグラフィックデータをカラーディスプレイ画面上の同じ場所に表示する際、どちらの色を優先して表示するかを決めます。）の指定ができます。

出力ポート：＄F5

■ グラフィックのカラー表示

1ドット単位で色の指定ができます。また、カラーディスプレイ上のグラフィックデータを白黒ディスプレイ画面へ重ね合わせて表示するか否かの指定ができます。

出力ポート：＄F7, ＄F6

■ バックグランドのカラー表示

カラーディスプレイ画面のバックグランドの色を指定できます。使用例として、カラーディスプレイ画面上にキャラクタを表示させずにグラフィックデータのみの表示とする場合、キャラクタとバックグランドの色を同色にし、プライオリティをグラフィックに指定します。

出力ポート：＄F4

詳細については、P[illegible]～P[illegible]の出力ポート＄F7～＄F4の項を参照ください。

オプションデバイスのセッティング

Chapter 3

この章では、MZ-2000の拡張オプションデバイスのセッティングの概要を解説しています。
オプションデバイスのうち次のものは、MZ-2000のメインキャビネット内にセッティングされ、本章の最初に取り付け方法が示されます。

MZ-1R01　グラフィックボード：コントローラ（ページ1…16KバイトRAM含む）
MZ-1R02　グラフィックメモリ：16KバイトRAM（ページ2またはページ3）
MZ-1U01　拡張I／Oポートユニット

その他のオプションデバイスはすべてMZ-1U01 拡張I／Oポートに各I／Oカードをセットし入出力機器とのインターフェイスをとります。本章の後半はその概要を示しています。取り扱いの詳細については各周辺機器のオペレーションマニュアルによってください。

3.1　グラフィックボードと拡張I/Oポートのセッティング

グラフィックボードMZ-1R01、グラフィックメモリMZ-1R02および拡張I/OポートMZ-1U01は、MZ-2000本体内の定められた取り付け場所に、正しくセッティングしなければなりません。

図3－1は、このオプションデバイスと、MZ-2000本体内で、それをセットすべき場所を示しています。

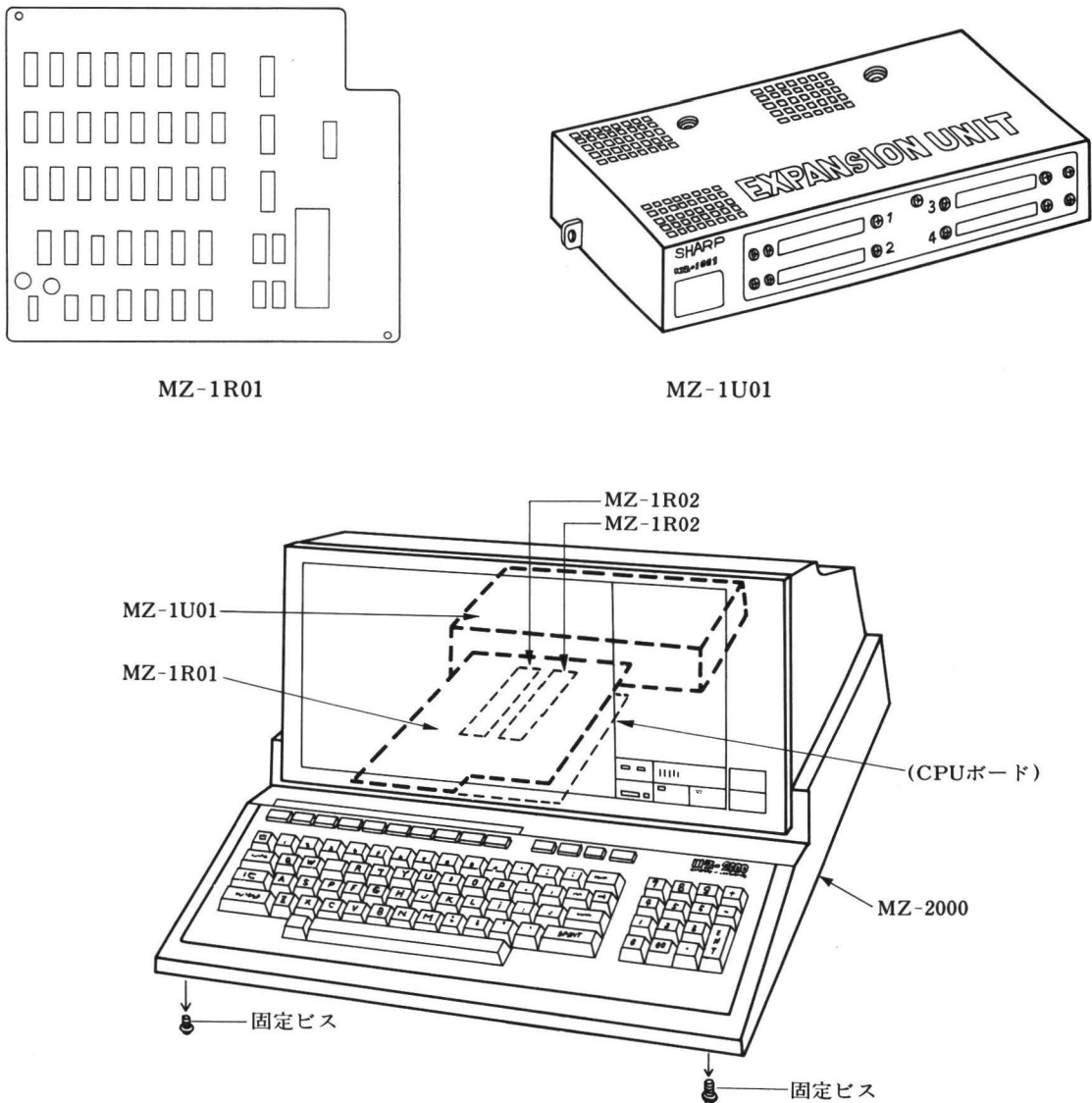



図　3−1　MZ-1R01、MZ-1U01とそのセッティング場所

これらのオプションデバイスをセットするために、はじめにMZ-2000の上部（CRTディスプレイ＋カセットテープデッキ部）を持ち上げなくてはなりません。その手順を次に示します。

(1)　MZ-2000の電源スイッチをOFFにし、電源プラグを外します。

(2)　MZ-2000キーボード部下側の固定ビス２本を図３－１に示すように外します。

(3) 固定ビスを外したら、本体上部を手前より静かに持ち上げ、図3－2に示すように支持アームを本体上部の底面から引き出し、下部のキャビネットに付けられているホールに入れて支えます。

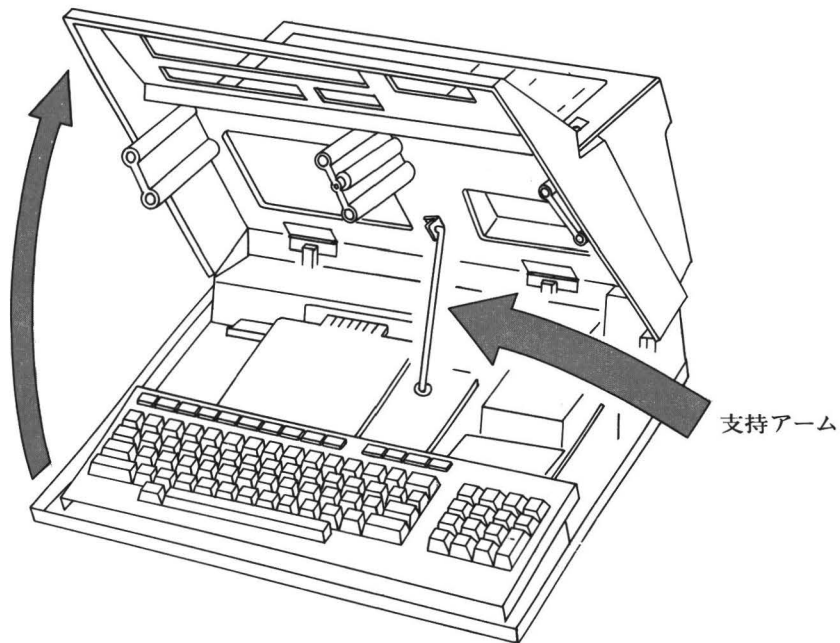



図 3－2 支持アームで支える

〔注 意〕 本体上部を持ち上げた状態で電源を入れてはなりません。電源を入れたまま接続操作を行うと回路部品を壊したり感電する危険性があります。
また、ビス等の金属類を本体内に落としたままにすると故障の原因となるので充分な注意が必要です。

3.1.1 グラフィックボードのセッティング

3.1.1.a グラフィックメモリのセッティング

グラフィックボードMZ-1R01には、グラフィックメモリ／ページ1が標準装備されています。このボードにページ2、3 (いずれも機種名はMZ-1R02) を拡張する場合は、グラフィックボード内の所定の位置に正しくセッティングをしなくてはなりません。

図3－3にグラフィックボード内のグラフィックメモリ／ページ2、3のセッティング場所を示します。

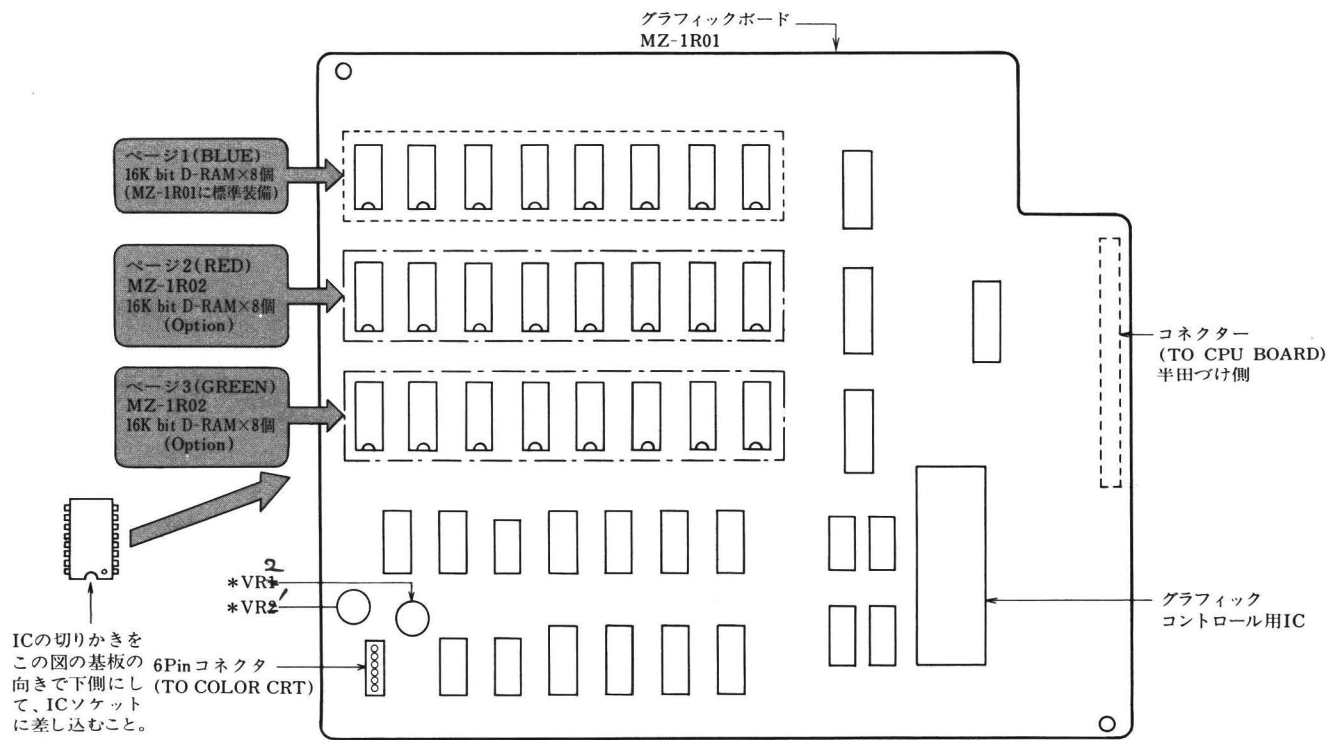



図 3－3

グラフィックメモリ(D-RAM)ICの装着方向は、すでに装着されているページ1のメモリICと同方向となるようにICソケットに挿入してください。(逆方向に装着しますと、ICが破壊しますので充分注意して行ってください。)

ページ2、ページ3ともメモリICを8個とも装着します。

グラフィックメモリを拡張しない場合 (ページ1のみで使用) は、この手順は不要です。

---

* VR1: カラーディスプレイ画面の垂直方向のポジション設定用の調整ボリューム
* VR2: カラーディスプレイ画面の水平方向のポジション設定用の調整ボリューム

上記2個のボリュームは、カラーディスプレイ MZ-1D01用に調整されていますので、このディスプレイ以外を使用する場合に限り再調整ください。

3.1.1.b グラフィックボードのセッティング

グラフィックボードは、CPUボードの真上に装着します。(図3－4) まずグラフィックボードをグラフィックボード固定用リブに挿入し(①)、つぎにグラフィックボードのコネクタAをCPUボードのコネクタBに差し込みます(②)。最後にカラーディスプレイ用DINコネクタからきているケーブルコネクタCをグラフィックボード上のコネクタD (6pin) に差し込みます(③)。これでグラフィックボードのセッティングは終了です。

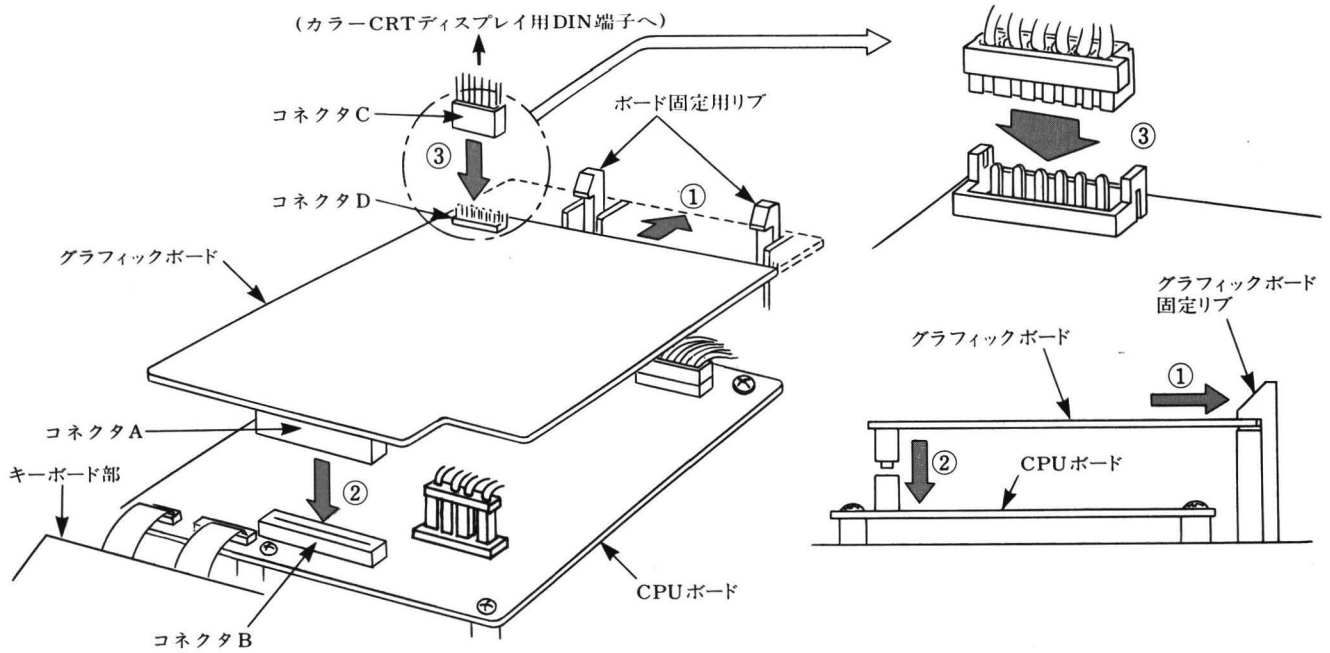



図 3—4

3.1.2 拡張I/Oポートのセッティング

MZ-2000の拡張I/OポートMZ-1U01は、前掲の図3－1に示されている通り、MZ-2000本体内の一番後部のスペースにセッティングを行います。この拡張I/Oポートのセッティング手順を図3－5、図3－6に示します。

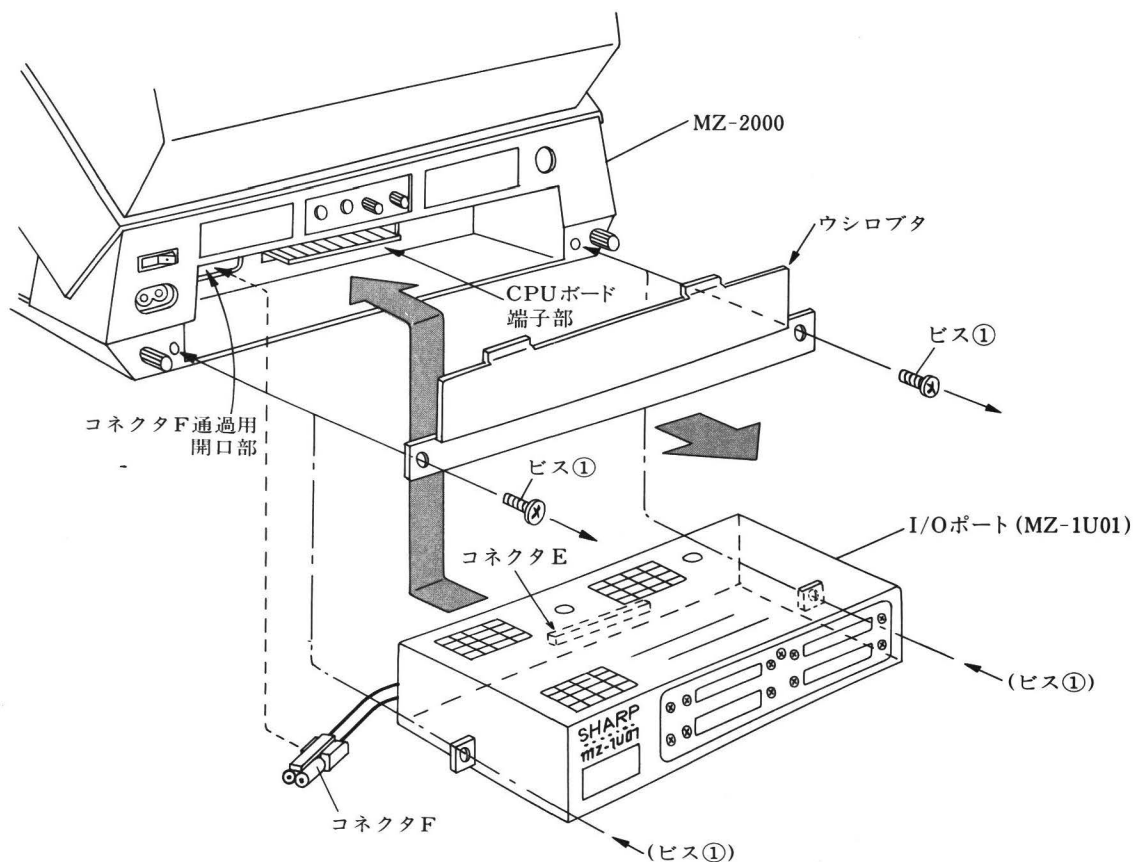



図 3−5

(1) MZ-2000本体後部にあるI/Oポート取付部のウシロブタの両サイドにある2本のビス①を外し、本体よりウシロブタを取り外します。

(2) この開口部へ拡張I/OポートMZ-1U01を図の向きに挿入します。この際、I/Oポートの電源用コネクタFを本体内部の電源ユニットへ接続できるように、図に示す開口部の奥の窓へ通しておきます。

(3) 開口部の奥にあるCPUボードの端子部とI/Oポートのマザーボードにあるコネクタ Eを接続します。

(4) (1)で取り外した2本のビス①で本体とI/Oポートを固定します。(以上の作業は図3－5を参照ください)

(5) I/Oポート電源用コネクタFを図3－6に示すように本体電源1次側基板上のコネクタGに差し込みます。

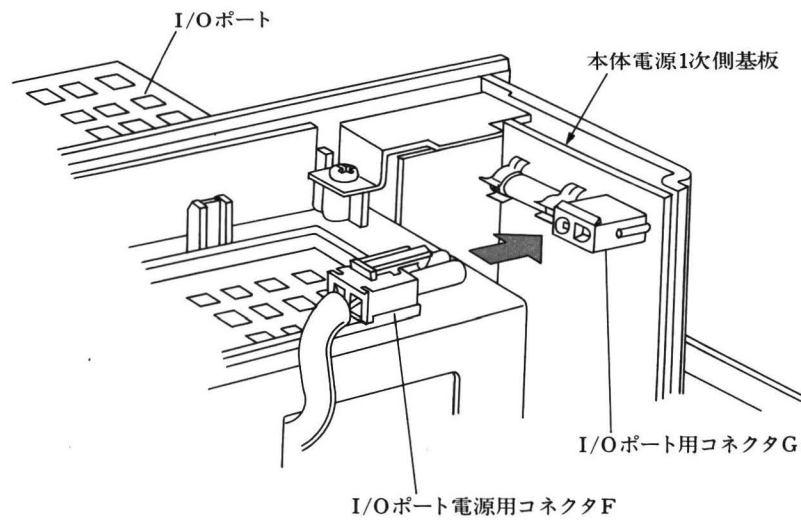



図　3－6

3.2　拡張I/Oポートへのオプションデバイスのセッティング

前節でセッティング方法を説明したデバイス以外のオプションデバイスは、全てそのインタフェイスカードを拡張I/Oポートの端子に差し込みます。

MZ-1U01は、4枚のインタフェイスカードをセッティングできるよう1番から4番までのスロットが用意されています。

プリンタインタフェイスカードはI/Oポートのスロット2番に、フロッピーディスクドライブインタフェイスカードはスロット4番にセッティングしてください。(各スロットの端子へは、まったく同一の信号が供給されていますが、カードの大きさの関係から、この2種類のカードのみスロットの場所を上記のごとく限定しています。)

その他のインタフェイスカードは、スロット1番か、スロット3番のいずれかにセッティングしてください。

図3－7にプリンタインタフェイスカードのセッティング方法を示します。

(1)　4本のビス②を外し、コネクタパネルをI/Oポートより取り外します。

(2)　2本のビス③を外し、コネクタブタをコネクタパネルより取り外します。

(3)　プリンタインタフェイスカードを部品側を上にしてI/Oポートのスロット2番に差し込みます。
　I/Oポート内には、それぞれインタフェイス用のレールガイドがありますので、これに沿って挿入します。
　(確実にI/Oポートのマザーボードにあるコネクタへ、このカードを差し込んでください。)

(4)　カードを正しくセットしたら、コネクタパネルを4本のビス②で元の位置に固定します。

図　3−7

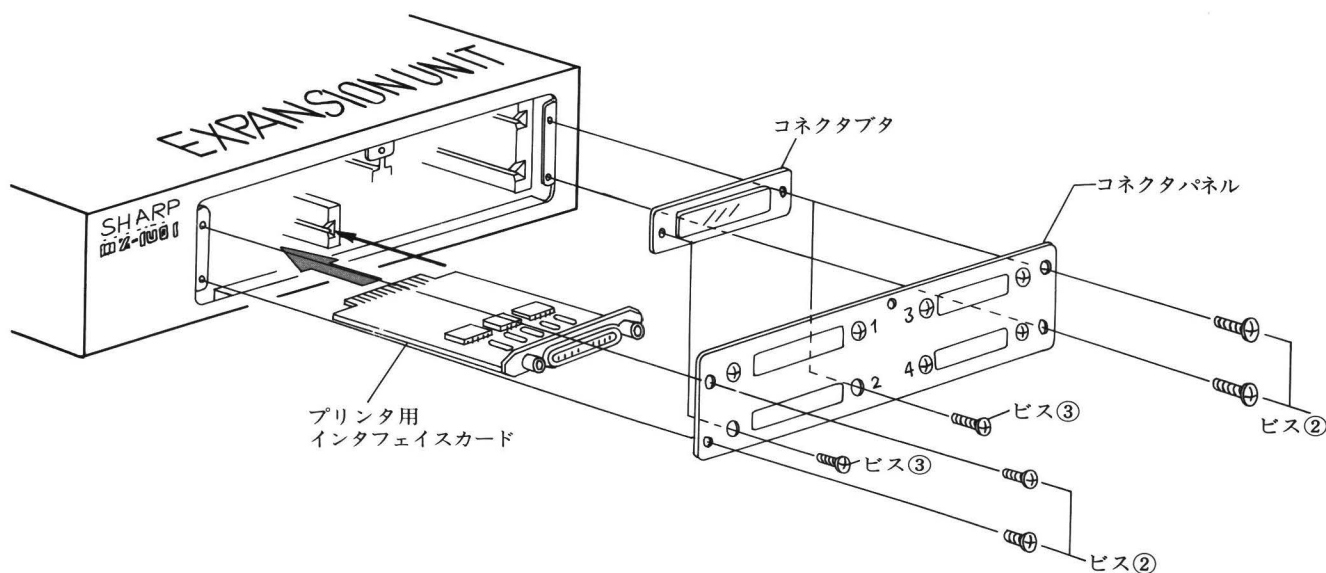
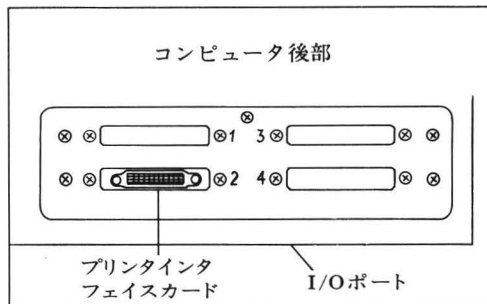
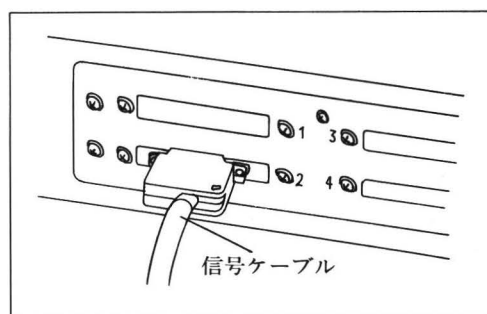



図3－8にインタフェイスカードと信号ケーブルの接続方法を示します。

(1) 右図は、プリンタインタフェイスカードをI/Oポートのスロット2番に装着した状態を示しています。

(2) プリンタの信号ケーブルをインタフェイスカードの信号ターミナルに接続します。ケーブルのコネクタは左右どちらを用いてもかまいませんが、接続の際、方向性がありますので注意してください。(逆方向の接続はできない構造になっています。)

(3) このコネクタの左右にある穴にケーブル付属のビスを通して、しっかりとビス止めし、コネクタを固定します。
このビス止めは、かならず行ってください(ビス止め2箇所)。
プリンタ側の接続については、各プリンタの取扱い説明書を参照の上、行ってください。

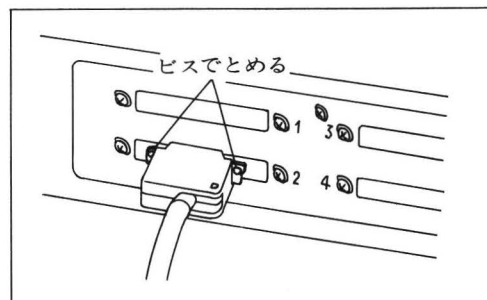



図 3−8

その他のオプションデバイスについても、スロット1、3または4番を用い、この節の説明に準じて正しくセッティングを行ってください。

MZ-2000のハードウェア構成

Chapter 4

この章は、MZ-2000のハードウェアを次の各セクションで概説しています。

■ MZ-2000システムダイアグラム

■ メモリ構成：メインメモリとIPL、V-RAMとの関係

■ 8255の信号系

■ 8253の信号系

■ PIOの信号系

■ MZ-2000本体およびグラフィックボード、拡張I/Oポートの全回路図

ただし、この章の内容は参考データであり、内容についてのお問い合わせには応じかねます。

4.1　MZ-2000のシステムダイアグラム

図4－1にパーソナルコンピュータMZ-2000内部のハードウェアシステム構成を示します。

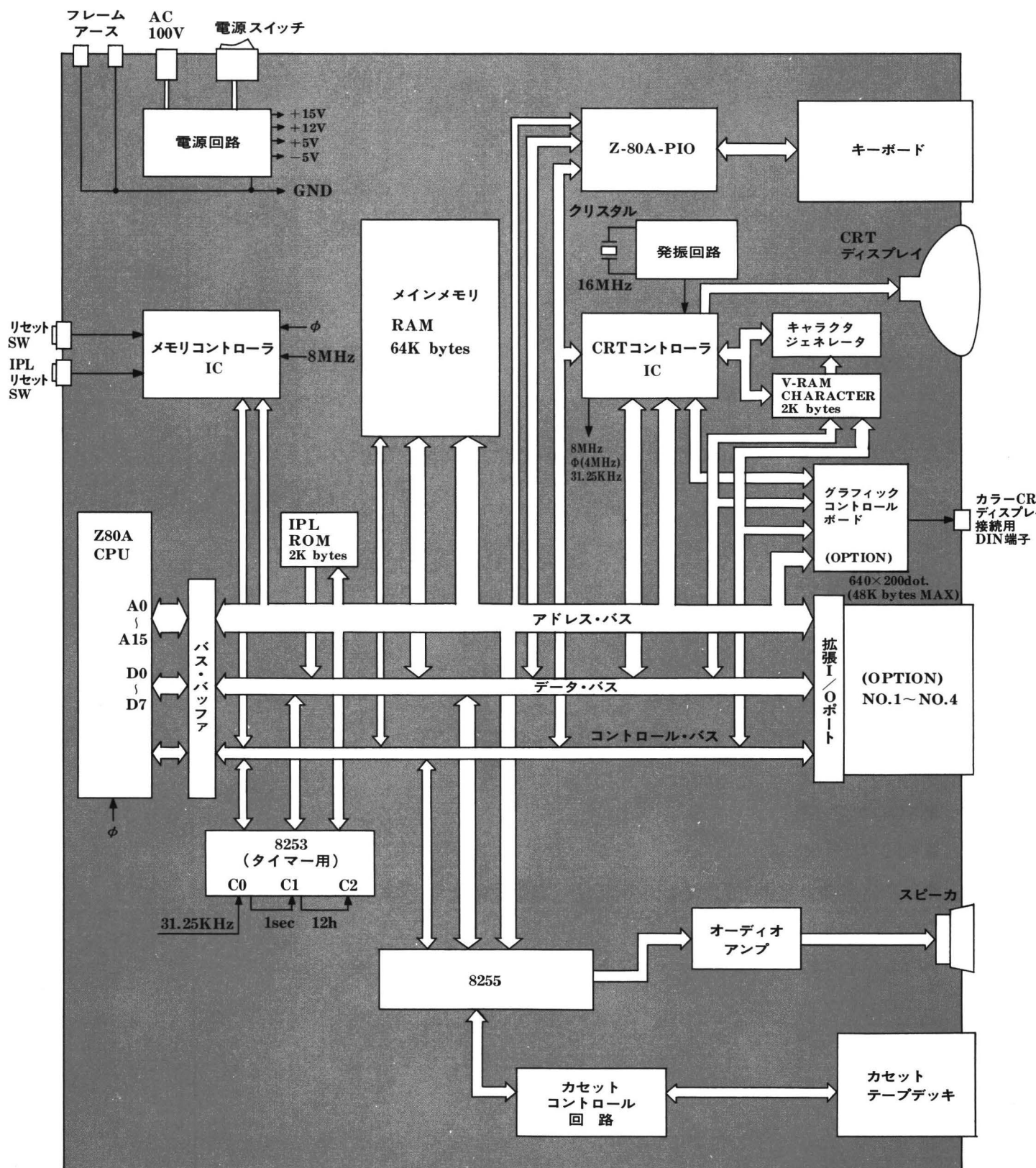



図　4－1　MZ-2000システム・ダイアグラム

4.2　メモリ構成

4.2.1　IPL動作時のメモリマップ

MZ-2000の電源スイッチをONにした時、あるいはIPLリセットスイッチを押した時もしくはプログラムによって(たとえばBASIC のBOOT コマンドの実行) 補助プログラムIPLがスタートする場合、MZ-2000は、図4－2に示すメモリ構成をとります。

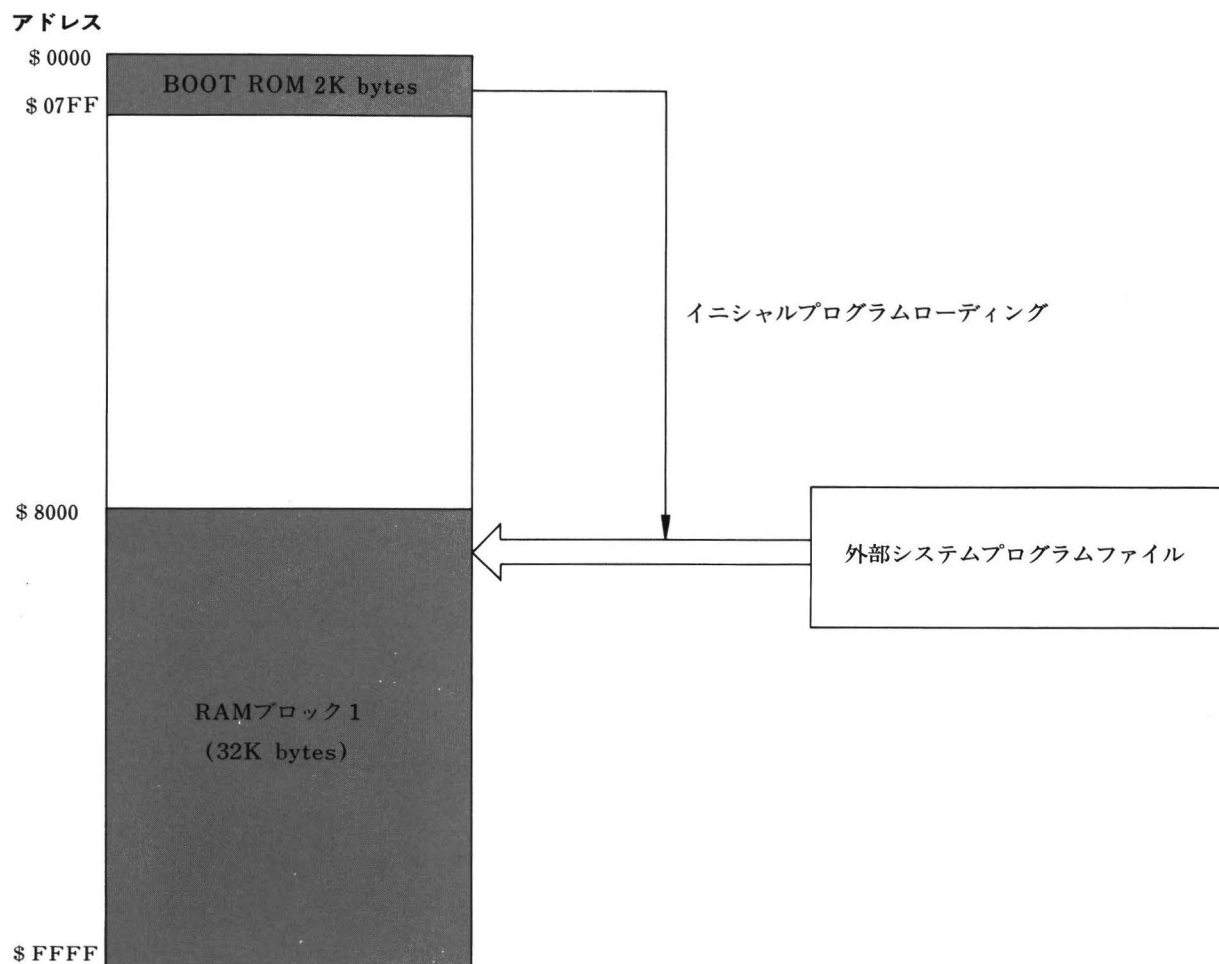



図　4－2　IPL動作時のメモリマップ

即ち、IPLプログラムを持つ2Kバイトの BOOT ROM がアドレス$0000～$07FFに、メインメモリ64Kバイトのうち前半の32KバイトのRAMエリア (以後RAMブロック1と呼ぶ) がアドレス $8000－$FFFF に設定されます(後半の32KバイトRAMエリアはCPUから切り離されていた状態にあります)。この状態でIPLはカセットテープファイルまたはディスケットファイルからアドレス$8000以降のRAMエリアへシステムソフトウェアのイニシャルローディングを実行します。

イニシャルローディングを終えるとシステムは、メモリ構成をノーマルステートに変え、そのアドレス$0000へコントロールを移します。

4.2.2　ノーマル時のメモリマップ

ＩＰＬによるシステムプログラムのイニシャルローディングを終了すると、MZ-2000は図４－３に示すメモリ構成をとります。

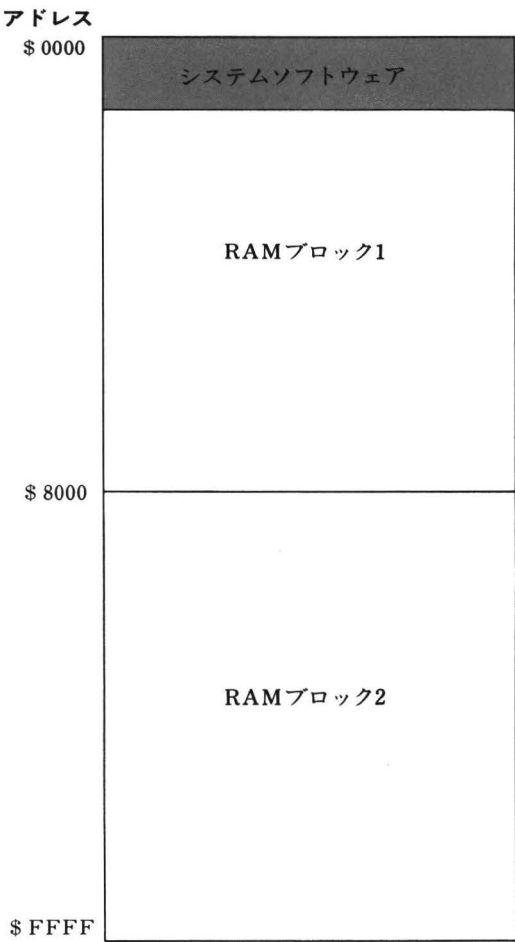



図　4－3　ノーマル時のメモリマップ

システムソフトウェアは、前記のＩＰＬ動作時にＲＡＭブロック１の先頭アドレス＄8000からロードされますが、ノーマルステートへのメモリ構成の変換によって、上図のように、システムソフトウェアがストアされたＲＡＭブロック１のアドレスは、＄0000～＄7FFFとなり、ＲＡＭブロック２が、＄8000～＄FFFF　の各アドレス空間を占めるようになります。このメモリ構成の変換直後にＣＰＵのリセットがかかってアドレス＄0000即ち、システムソフトウェアの先頭からプログラムが実行されます。

4.2.3 V-RAMアクセス時のメモリマップ

メモリアドレスの切り替えは、V-RAMアクセス時にも行なわれます。

V-RAMアクセス時のメモリアドレスの切り替えはキャラクタまたはグラフィックのいずれかを選択することによって行なわれその際のメモリマップは、それぞれ異なります。

すなわち、V-RAMキャラクタを選択した場合、図4－4に示すようにRAMブロック3の4Kバイト($D000～$DFFF)をアドレッシング空間からはずして、そのかわりにV-RAMをアクセス状態とします。したがってもし表示プログラムまたはデータが、RAMブロック3にある場合は、コントロールができないことになります。V-RAMグラフィックを指定した場合、図4－5に示すようにRAMブロック3の16Kバイト($C000～$FFFF)をアドレッシング空間からはずし、V-RAMをアクセス状態とします。

メインメモリとV-RAMのアドレス切り替えはPIOのA7ポートによって実行され、またキャラクタとグラフィックの切り替えは同じくPIOのA6ポートによって実行されます。さらにグラフィックRAMのページ1、ページ2及びページ3の切り替えはOUTポート$F7によって決められます。

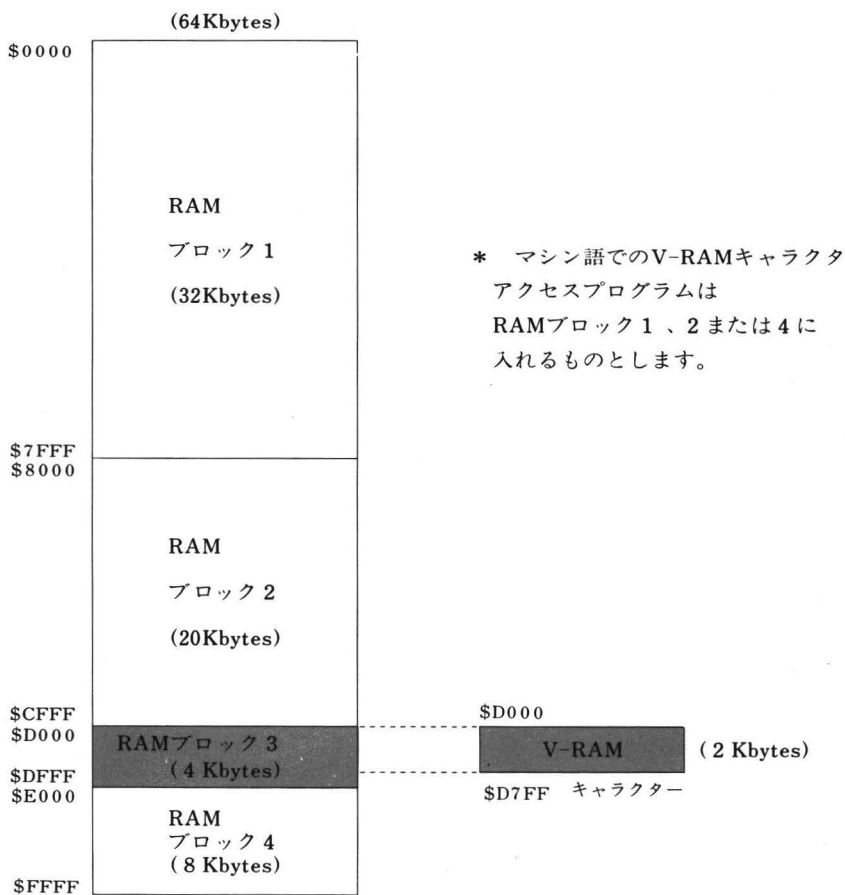



図 4-4 メインメモリとV-RAM(キャラクタ)のアドレス切り替え

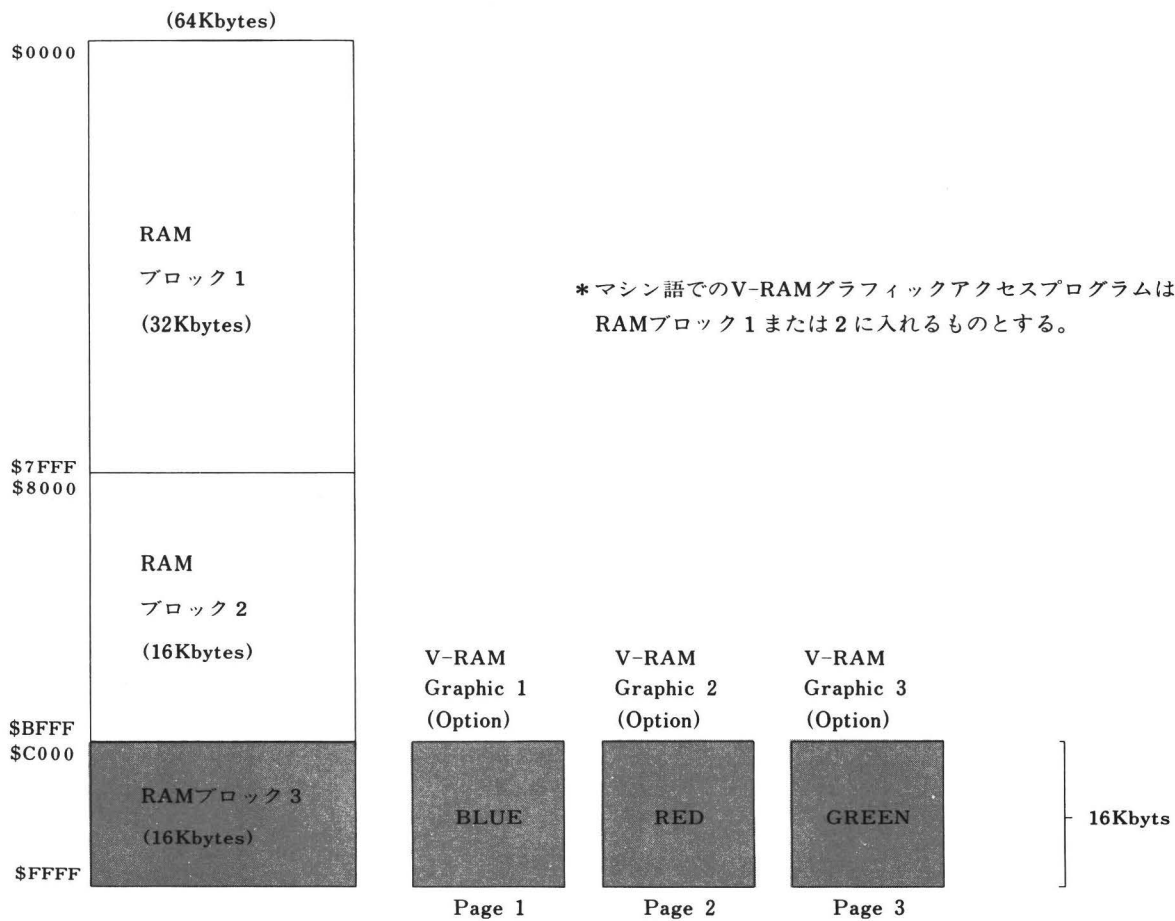



図 4-5 メインメモリとV-RAMグラフィックのアドレス切り替え

V-RAMグラフィックの名称について

このマニュアル内でのV-RAMグラフィックの各名称は標準BASIC使用時にはおのおの下記名称に対応します。(標準BASICとはBASICインタープリタMZ-1Z001およびMZ-2Z001のことです。)

マニュアル内での名称	標準BASIC使用時の名称
V-RAMグラフィックBLUE	V-RAMグラフィック1　(ページ1)
〃　RED	〃　2　(ページ2)
〃　GREEN	〃　3　(ページ3)

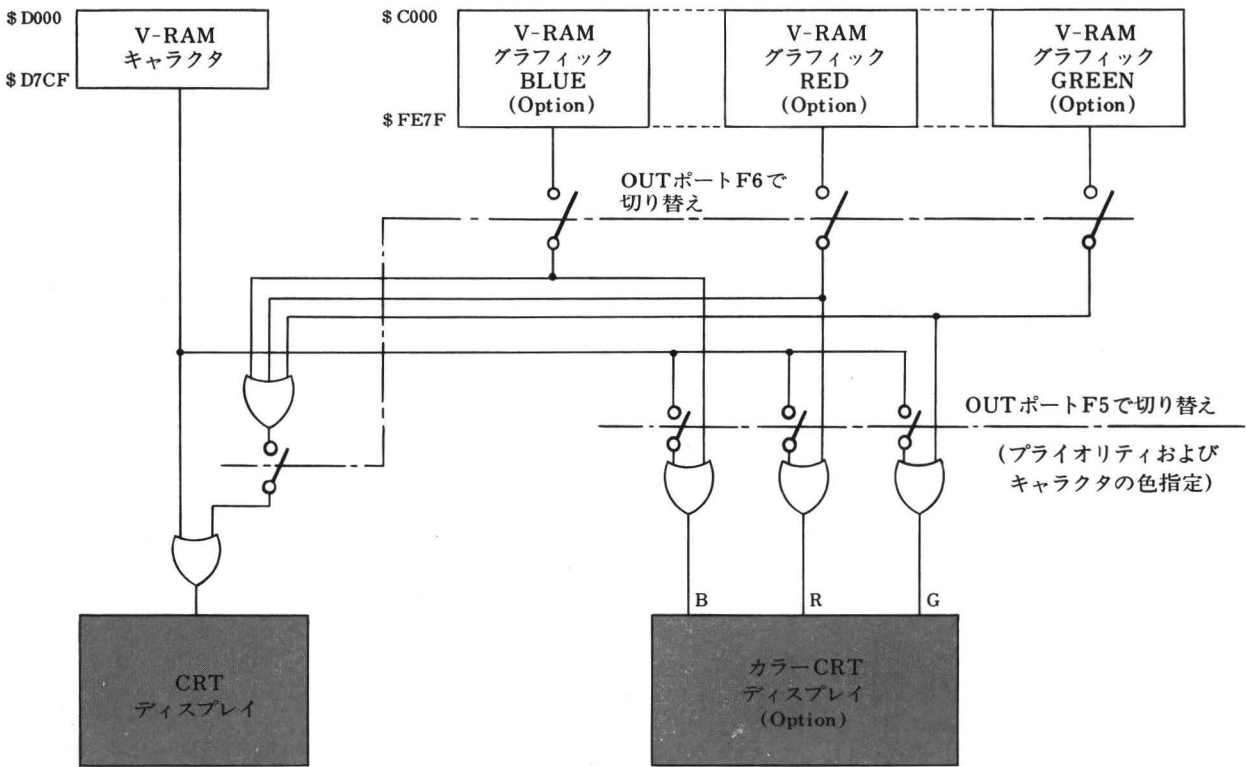
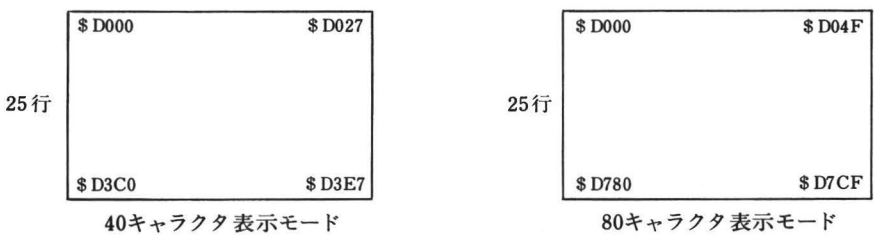
V-RAM（キャラクタ、グラフィック）の出力とCRTディスプレイのつながりを図4－6に示します。

注〕 プライオリティとは、カラーCRTディスプレイ上にキャラクタとグラフィックの重ね合わせ表示を行なう場合、両データを同じ場所に表示する際、どちらのデータを優先して表示するかを決めるものです。

図 4－6 V-RAMとCRTディスプレイのつながり

V-RAMアドレスとCRTディスプレイ上の該当位置との関係を図4－7に示します。

〈1〉 V-RAMキャラクタ

〈2〉 V-RAMグラフィック

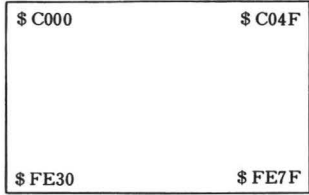



図 4－7 V-RAMアドレスとCRTディスプレイ

グラフィック用V-RAMの入出力制御は、OUTポート＄F4、＄F5、＄F6および＄F7に出力するデータによっておのおの、次のように制御することができます。(表中の○は可を、×は不可を示します。)

〈ポート＄F7〉

グラフィックエリアの切り替え

表 4-1a

ポート＄F7への出力データ	V-RAM(BLUE)	V-RAM(RED)	V-RAM(GREEN)
00	×	×	×
01	○	×	×
02	×	○	×
03	×	×	○

F7(16進)＝247(10進)

ここで　○　V-RAM転送(データの読み出し／書き込み)可

　　　　×　V-RAM転送(データの読み出し／書き込み)不可

V-RAM（グラフィック）に対して転送を行う場合、同時に2つ以上のV-RAMを指定するとハード上の破壊を生じる場合があるので、上表のよう、かならずグラフィックエリアのいずれか1つのV-RAMのみを選択するように配慮されています（したがってソフト上誤って、このポートにデータ04を与えても、B・G 転送可とはならずR・G・B共転送不可の状態となります)。

(具体例) ＄F7ポートに02Hを出力して、＄C000に01Hを格納したとすると、V-RAMグラフィック（RED）上の＄C000に01Hが転送されます。

〈ポート＄F6〉

V-RAMグラフィックデータの白黒画面への重ね合わせ表示およびR・G・Bのディスプレイ表示選択

表 4-1b

<table>
<tr><th rowspan="2">ポート＄F6への出力データ</th><th rowspan="2">グラフィックデータの白黒画面への重ね合わせ</th><th colspan="3">V-RAMグラフィックのディスプレイ表示</th></tr>
<tr><th>BLUE</th><th>RED</th><th>GREEN</th></tr>
<tr><td>00</td><td rowspan="8">○</td><td>×</td><td>×</td><td>×</td></tr>
<tr><td>01</td><td>○</td><td>×</td><td>×</td></tr>
<tr><td>02</td><td>×</td><td>○</td><td>×</td></tr>
<tr><td>03</td><td>○</td><td>○</td><td>×</td></tr>
<tr><td>04</td><td>×</td><td>×</td><td>○</td></tr>
<tr><td>05</td><td>○</td><td>×</td><td>○</td></tr>
<tr><td>06</td><td>×</td><td>○</td><td>○</td></tr>
<tr><td>07</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td>08</td><td rowspan="8">×</td><td>×</td><td>×</td><td>×</td></tr>
<tr><td>09</td><td>○</td><td>×</td><td>×</td></tr>
<tr><td>0A</td><td>×</td><td>○</td><td>×</td></tr>
<tr><td>0B</td><td>○</td><td>○</td><td>×</td></tr>
<tr><td>0C</td><td>×</td><td>×</td><td>○</td></tr>
<tr><td>0D</td><td>○</td><td>×</td><td>○</td></tr>
<tr><td>0E</td><td>×</td><td>○</td><td>○</td></tr>
<tr><td>0F</td><td>○</td><td>○</td><td>○</td></tr>
</table>

F6(16進)＝246(10進)

(具体例) ＄F6ポートに06Hを出力したとすると、カラーCRTディスプレイ上にV-RAMグラフィック(BLUE)のデータは表示されず、V-RAMグラフィック(RED)、(GREEN)のデータがディスプレイ表示され、その表示されたデータが白黒CRT画面上にキャラクタデータと重ね合わせて表示されます。

〈ポート＄F5〉

キャラクタデータのカラーCRT画面上への重ね合わせ表示、キャラクタデータの色調選択およびキャラクタとグラフィックの両データが表示画面上重なった時に、どちらの色調を優先するかを選択（プライオリティー）…第4－6図の㈲参照。

表 4-1c

ポート＄F5への出力データ	プライオリティー	キャラクターの色調			色調
		BLUE	RED	GREEN	
00	キャラクタ	×	×	×	黒
01		○	×	×	青
02		×	○	×	赤
03		○	○	×	紫
04		×	×	○	緑
05		○	×	○	水色
06		×	○	○	黄
07		○	○	○	白
08	グラフィック	×	×	×	黒
09		○	×	×	青
0A		×	○	×	赤
0B		○	○	×	紫
0C		×	×	○	緑
0D		○	×	○	水色
0E		×	○	○	黄
0F		○	○	○	白

F5（16進）＝245（10進）

㈲ プライオリティをグラフィック側に選択し、バックランドとキャラクタの色調選択を同一にするとカラーCRT画面上へのキャラクタデータの表示はされず、グラフィックデータのディスプレイ表示のみとなります。

〈ポート＄F4〉

カラーCRT画面のバックグランドの色調選択

表 4-1d

ポート＄F4への出力データ	バックグランドの色調			色調
	BLUE	RED	GREEN	
00	×	×	×	黒
01	○	×	×	青
02	×	○	×	赤
03	○	○	×	紫
04	×	×	○	緑
05	○	×	○	水色
06	×	○	○	黄
07	○	○	○	白

F4（16進）＝244（10進）

4.3　8255，8253およびPIOの各信号系

この節では、回路中で重要な役割りを果たしているコントローラ、8255、8253、PIO についてそのコントロール内容と信号系を解説しています。各ＩＣは次に示すように、CPU の入出力ポート＄E0 ～＄EB を介してアクセスされます。

表　4—2

CPUの入出力ポート	コントローラ	各ポートのサービスモード
＄E 0	8255	P_A：出力
＄E 1		P_B：入力
＄E 2		P_C：出力
＄E 3		モードコントロール
＄E 4	8253	C_0：モード２（16ビット・レートジェネレータ）
＄E 5		C_1：モード２（16ビット・レートジェネレータ）
＄E 6		C_2：モード２（16ビット・レートジェネレータ）
＄E 7		モードコントロール
＄E 8	ＰＩＯ	A：出力モード３（ビットコントロール）
＄E 9		モードコントロールA
＄E A		B：入力モード３（ビットコントロール）
＄E B		モードコントロールB

4.3.1 8255まわりの信号系

8255 (programmable peripheral interface) は、オートカセットコントロール、CRTディスプレイのリバース、ブランクコントロール、ブート／ノーマルのメモリ切替え、サウンド発生用ソースパルスの出力のそれぞれ制御を行っています。

図4－8は8255まわりの信号系をまとめたものです。表4－3に各ポートのコントロール内容を示します。

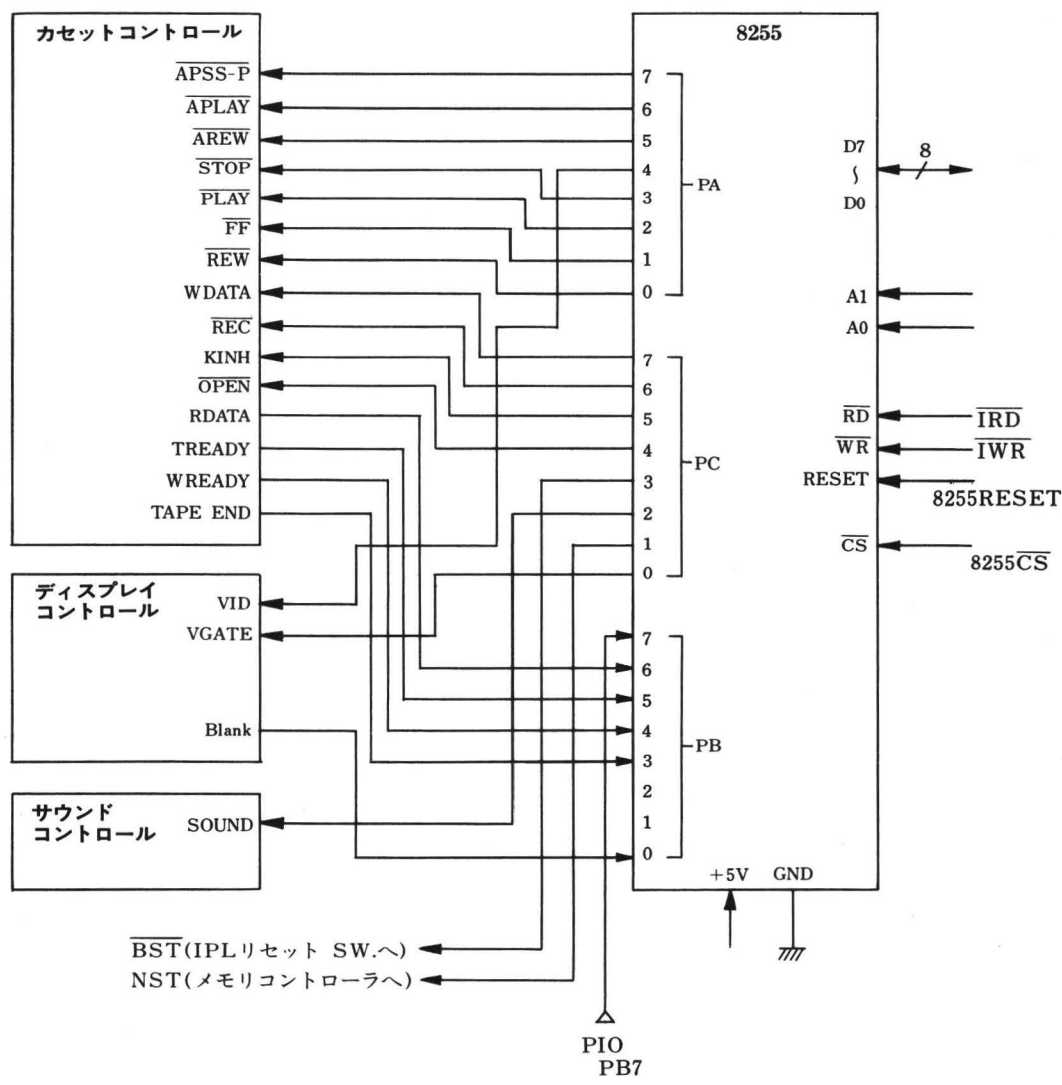



図　4－8　8255まわりの信号系

表 4-3

ポートA （ポートアドレス＝$ E0）

ポート端子	アクティブ	コントロール内容		信号名
PA_7	L	プログラム頭出し(APSS)の準備(カセット)	＊1	$\overline{\text{APSS-P}}$
PA_6	L	巻き戻し終了時の自動再生　　(カセット)	＊2	$\overline{\text{APLAY}}$
PA_5	L	再生(録音)時の自動巻き戻し　(カセット)	＊3	$\overline{\text{AREW}}$
PA_4	L	表示画面全体のB/Wを反転させる(ディスプレイ)		VID
PA_3	L	カセットの動作停止　(‾|_|‾)		$\overline{\text{STOP}}$
PA_2	L	カセットのPLAY　(‾|_|‾)		$\overline{\text{PLAY}}$
PA_1	L	カセットのFF　(‾|_|‾)		$\overline{\text{FF}}$
PA_0	L	カセットのREW　(‾|_|‾)		$\overline{\text{REW}}$

ポートB （ポートアドレス＝$ E1）

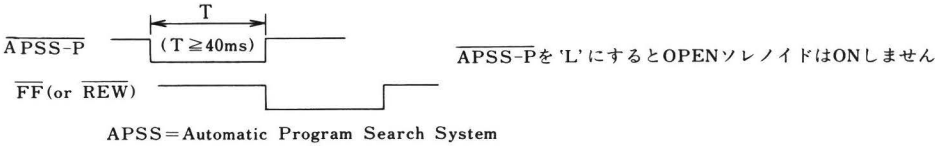
ポート端子	アクティブ	コントロール内容	信号名
PB_7	~~H~~ L	カセットPLAY中の [BREAK] キーの検出	−
PB_6	−	カセット読み出しデータ	RDATA
PB_5	L	カセットがセットされている状態	TREADY
PB_4	L	書き込み可能状態(ライトプロテクト用リブ有り)(カセット)	WREADY
PB_3	L	テープ動作中信号　(カセット)	TAPE END
PB_2	−	−	−
PB_1	−	−	−
PB_0	H	ディスプレイのブランキング期間	Blank

ポートC （ポートアドレス＝$ E2）

ポート端子	アクティブ	コントロール内容	信号名
PC_7	−	カセットへの書き込みデータ	WDATA
PC_6	L	REC待機(PLAY ‾|_|‾ でWRITE状態)	$\overline{\text{REC}}$
PC_5	H	FF、REW、STOPの各キー入力を禁止	KINH
PC_4	L	EJECT動作(‾|_|‾)	$\overline{\text{OPEN}}$
PC_3	L	IPLスタート	$\overline{\text{BST}}$
PC_2	−	サウンド発生用ソースパルス出力	SOUND
PC_1	H	メモリをノーマル状態とし＄0000スタート	NST
PC_0	H	ディスプレイを強制的にブランクします。	VGATE

補足説明

＊1　$\overline{\text{APSS-P}}$　FF、REW以外のモードの際にこの入力を'L'とする(約40ms)→PLAYソレノイドがONする（テープ・セット状態でなければONしない）→$\overline{\text{FF}}$又は$\overline{\text{REW}}$を'L'にするとAPSSモードとなります(ヘッドとテープがタッチ、ピンチローラとテープはタッチしない状態にて、FF又はREWとなり所望のファイルをす早く探し出すことができます。

APSS＝Automatic Program Search System

＊2　$\overline{\text{APLAY}}$　H……テープ巻き戻し終了時にストップ状態となります。
L……テープ巻き戻し終了時に一度ストップ状態となり、自動的に再生(PLAY)状態となります。

＊3　$\overline{\text{AREW}}$　H……テープ再生(PLAY)又は録音中にテープエンドにきた時ストップ状態となります。
L……テープ再生(PLAY)又は録音中にテープエンドにきた時一度ストップ状態となり、自動的に巻き戻し(REW)状態となります。
早送り(FF)終了時は$\overline{\text{AREW}}$に無関係にストップ状態となります。

4.3.2 8253まわりの信号系

8253 (programmable interval timer) は、そのカウンタ＃０、＃１、＃２で内蔵時計の役割りを果しています。
カウンタ＃０、＃１、＃２とも mode 2：レート・ジェネレータ (rate generator) として使用し、いずれも16bit バイナリカウンタです。

カウンタ＃０は、31.25 kHzの入力パルスをカウントし１sec 毎にOUT０にパルスを出力します。カウンタ＃１はそのパルスをカウントして、12h毎OUT１にパルスを出力します。カウンタ＃２は、そのパルスをカウントして０、１を繰り返し、AM/PM のフラグの役割りを果たします。図４－９参照。

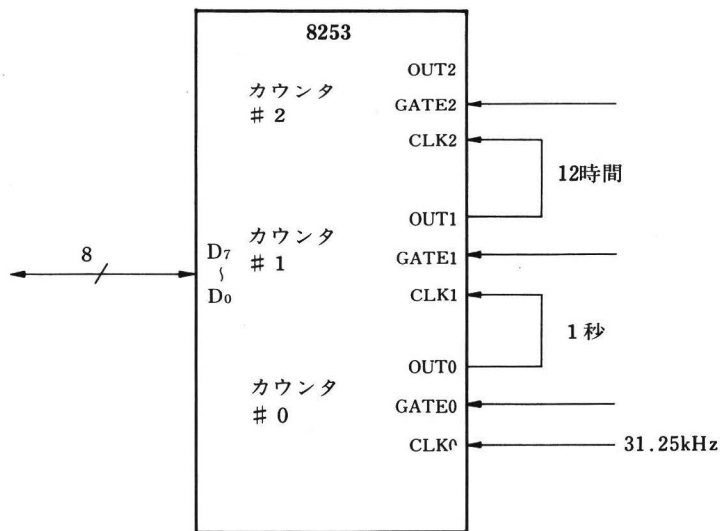



図 4−9 8253まわりの信号系

4.3.3 PIOまわりの信号系

Z80A-PIO (parallel input/output interface controller) は、キー ボードスキャン用ストローブ信号の出力、キーデータの入力、キーストローブをlow にする操作、およびV-RAMのアドレス切替え、40キャラ/80キャラモード切替えコントロール信号の出力を行います。

図4 —10は、PIO まわりの信号系をまとめたものです。次表に各ポートのコントロール内容を示します。

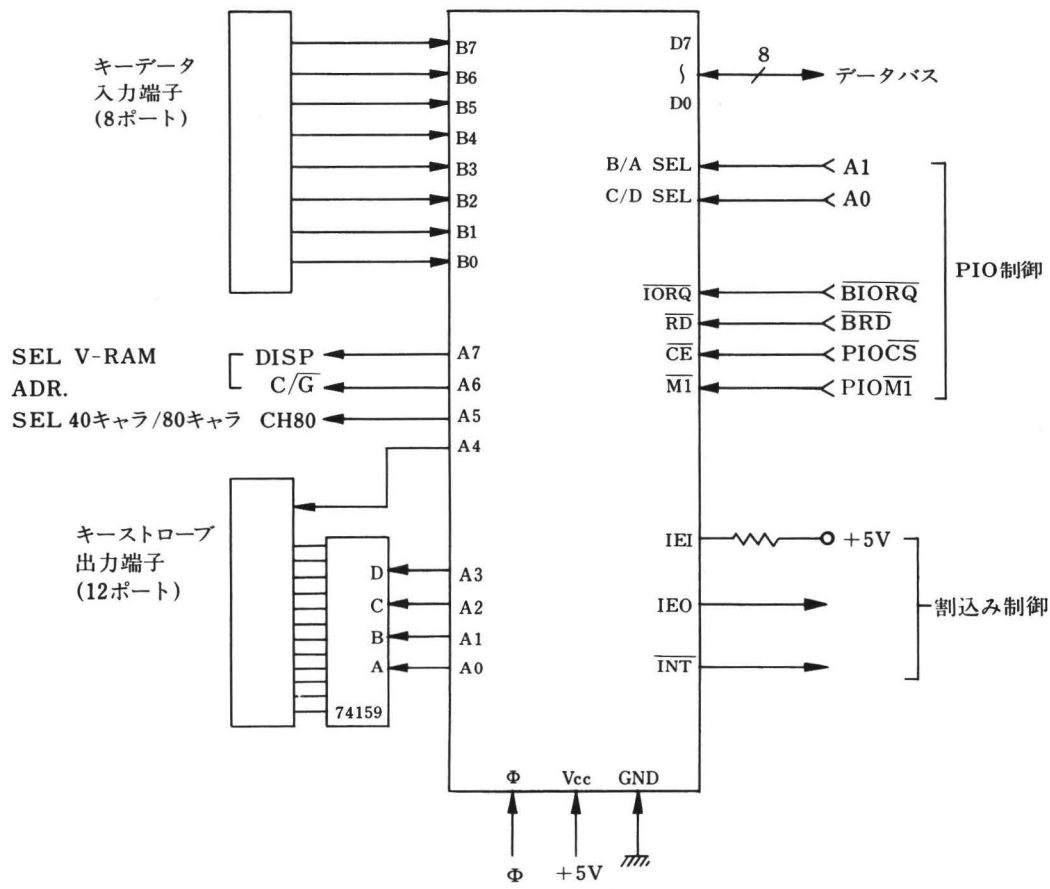



図 4—10 PIOまわりの信号系

表 4-4

ポートA (ポートアドレス＝＄E8)

ポート端子	アクティブ	コントロール内容	信号名
A_7	H	(キャラクタ＄D000～＄D7FF、グラフィック＄C000～＄FFFF)のアドレスをV-RAMに切替えます。	DISP
A_6	H	V-RAMキャラクタをアクセスします(L：V-RAMグラフィックをアクセス)	$C/\overline{G}$
A_5	H	画面を80キャラクタモードとします(L：40キャラ)	CH80
A_4	L	キーストローブ信号を全てLにします。	—
A_3		キーボードスキャン用ストローブ信号の出力	—
A_2			—
A_1			—
A_0			—

ポートB (ポートアドレス＝＄EA)

ポート端子	アクティブ	コントロール内容	信号名
B_7		キーボードスキャン用データ入力(負論理)	
B_6			
B_5			
B_4			
B_3			
B_2			
B_1			
B_0			

キーボードスキャンのストローブ信号とビットデータの関係を表4-5に示しています。

キーストローブ信号はPIOのポートAのA_3～A_0から出力され、マルチプレクサ159によってキーマトリクスの12端子に順番に出力されます。表の右端の欄がそのキーストローブ0H～BHを分類しています。(CH～FHは無効)

PIOのポートBに入力されるビットデータが列に示されています。

たとえばキーストローブ"6"のとき、ビットデータが"F7H"であれば、ビット3に対応するキー、即ち S キーが押されていることが判別されます。

表 4-5 キースキャニングのストローブとビットデータ

Eレジスタ上位4ビット	キーモード数	ビットデータの各ビット(負論理) 0	1	2	3	4	5	6	7	キーストローブ
0	1	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	0
		F_9	F_{10}	8	9	00	.	+	−	1
1		0	1	2	3	4	5	6	7	2
		TAB	SPACE	CR	↑	↓	←	→	BREAK	3
2	3 ～ 6	／ ← メ →	A a チ □	B b コ ♣	C c ソ ♥	D d シ □	E e イ ィ □	F f ハ □	G g キ □	4
		H h ク □	I i ニ □	J j マ □	K k ノ □	L l リ □	M m モ ¥	N n ミ □	O o ラ □	5
3		P p セ □	Q q タ □	R r ス □	S s ト □	T t カ □	U u ナ □	V v ヒ ◆	W w テ □	6
		X x サ ♠	Y y ン □	Z z ツ ッ □	^ ~ ヘ	＼ | ヲ	? ↑ ロ ↓	. > ル 。	, < ネ π	7
4	2 ～ 4	0 ￣ ワ	1 ! ヌ	2 " フ	3 # ア ァ	4 $ ウ ゥ	5 % エ ェ	6 & オ ォ	7 ' ヤ ャ	8
		8 (ユ ュ	9) ヨ ョ	: * ケ	; + レ	− = ホ	@ 、 ゛	[{ ゜ 「		9
5		] } ム 」		CLR HOME	INST DEL					A
	特殊キー	GRPH	SFT LOCK	SHIFT	カナ					B

■ キー割り込みを行うために

PIOをプログラムすることによってキー割り込みコントロールを行うことができます。次に示すプログラム例は BREAK キーが押された時、CPUに割り込みをかけて、コントロールをブレーク割り込み処理ルーチンへ移してやるものです。ここで各条件は、次のそれぞれの値を例にとってみることにします。

PIO……………ビット制御モード（モード3）とする

CPU割り込みモード………モード2、ベクトル割り込み（従ってIレジスタには割り込み処理ルーチンのアドレステーブルの上位8ビットがセットされなければならない。）

メモリ上のブレーク割り込み処理ルーチンのスタートアドレス………＄5080番地

メモリ上のブレーク割り込み処理ルーチンのスタートアドレステーブル………＄3370番地（従って＄3370番地に80H、＄3371番地に50Hがストアされなければならない）

```
LD  A, 33H          } CPUの割り込みベクトルレジスタＩを33Hとする
LD  I, A            }
IM  2               } CPUの割り込みモード２を設定する
LD  HL, 5080H       } メモリ上のブレーク割り込み処理ルーチンのアドレステーブル＄3370番地に、アドレス＄
LD  (3370H)、HL     } 5080番地をセットする
LD  A, 70H          } PIOに割り込みベクトル70H（＄3370の下位８ビット）をセットする（転送データ70Hの
OUT (EBH), A        } 最下位ビットが"０"であるのでこの設定が行われる）
LD  A, CFH          } PIOのポートＢをモード３（ビット制御モード）とする
OUT (EBH), A        }
LD  A, FFH          } PIOのポートＢをすべて入力ポートとする
OUT (EBH), A        }
LD  A, 97H          } 割り込みイネーブル・フリップ・フロップをセットし割り込み制御語を設定する
OUT (EBH), A        }
LD  A, 7FH          } BREAK キーのビットデータ"７"をマスクとしてセットする
OUT (EBH), A        }
IN  A, (E8H)        }
AND F0H             } キーストローブ信号を"３"にセットする。ストローブゲートはPIOのＡ４ポートを"H"
OR  13H             } にすることによってオープンされる
OUT (E8H), A        }
```

上記のモードセッティングを実行した後は、 BREAK キーを押すことによってテーブル＄3370番地が参照され、コントロールが、＄5080番地へ移されます。ブレーク割り込み処理を終えて、割り込みがかけられた箇所からメインプログラムを実行再開するには、RETI命令を実行させればよいことになります。

4.4　MZ-2000回路図

この節ではユーザの参考のため、MZ-2000本体と、グラフィックボード　、拡張I／Oポートの全回路図を次の順に掲載しています。

■MZ-2000本体の回路図

〔1〕 CPU ボード・ブロック1：CPU信号系
〔2〕 CPU ボード・ブロック2
〔3〕 CPU ボード・ブロック3：8255、PIO 信号系
〔4〕 CPU ボード・ブロック4：RAM ブロック
〔5〕 CPU ボード・ブロック5 (各端子表)
〔6〕 CRT ディスプレイコントロール部
〔7〕 カセットテープデッキコントロール部
〔8〕 電源回路

■ グラフィックボードMZ-1R01の回路図

〔1〕 コントロール部
〔2〕 RAMブロック

■ 拡張I／Oポートユニット、MZ-1U01の回路図

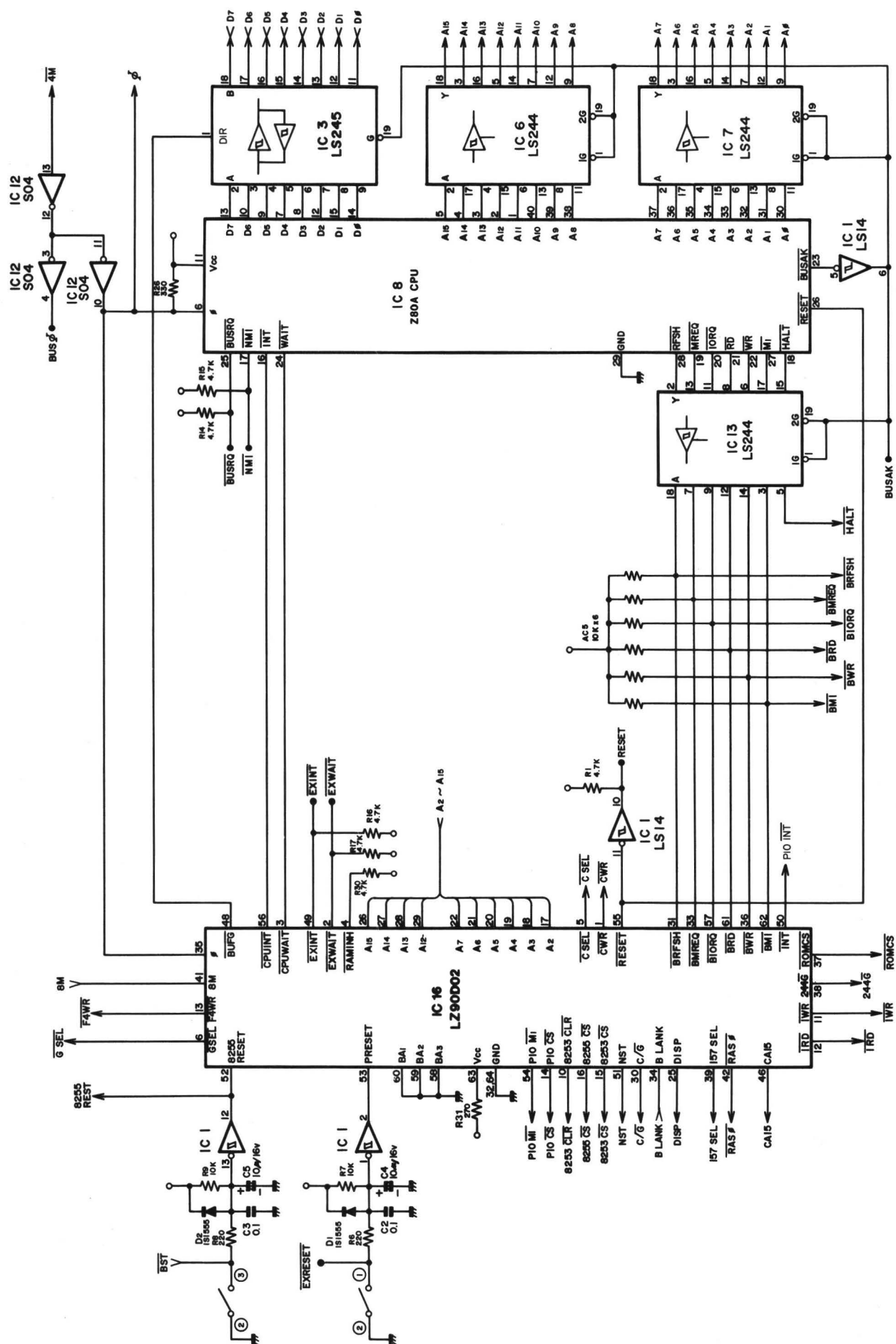



図　4-11　CPUボード〔1〕：CPU信号系

図 4-12 CPUボード〔2〕

図 4-13 CPUボード〔3〕：8255, PIO信号系

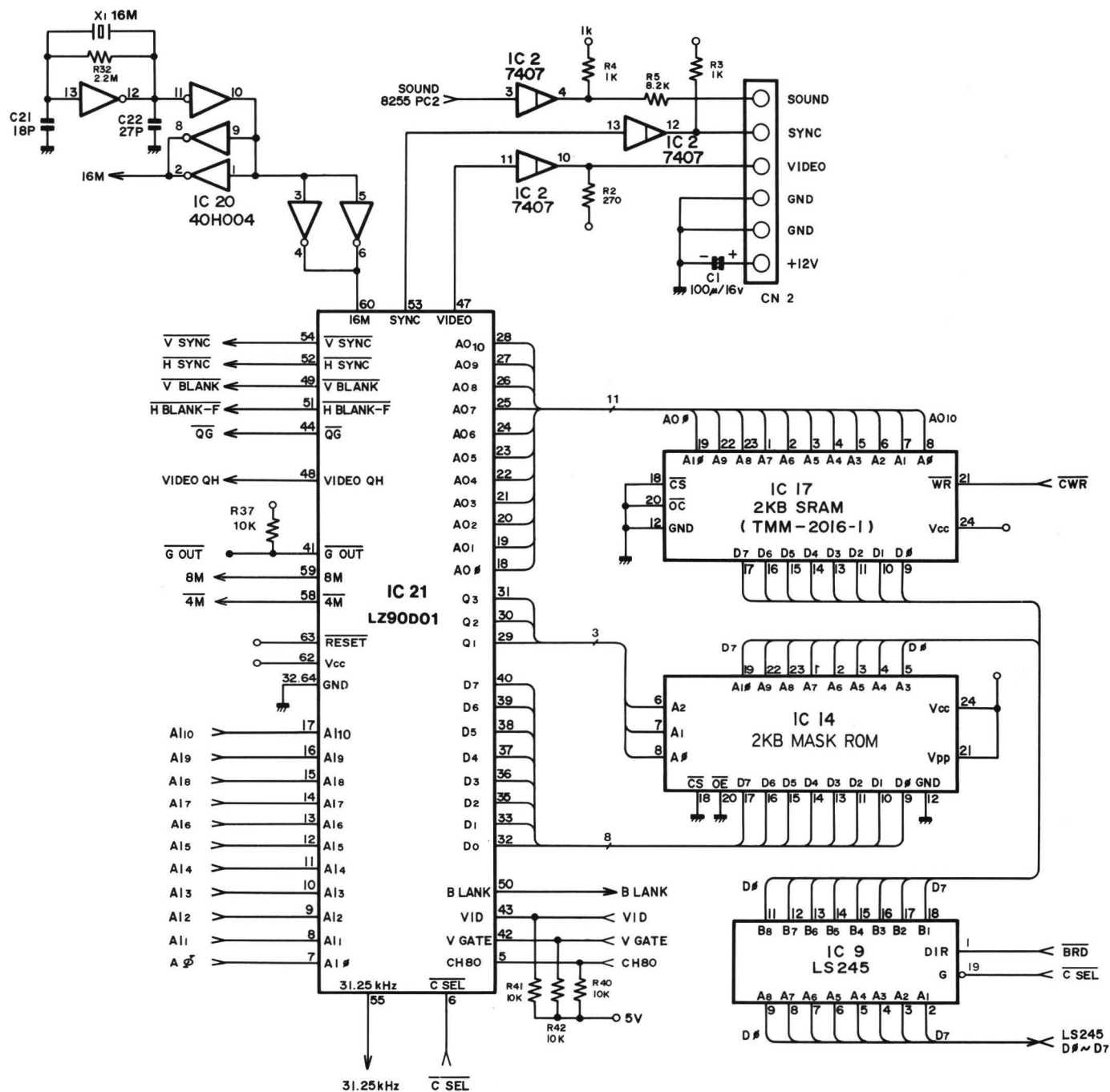
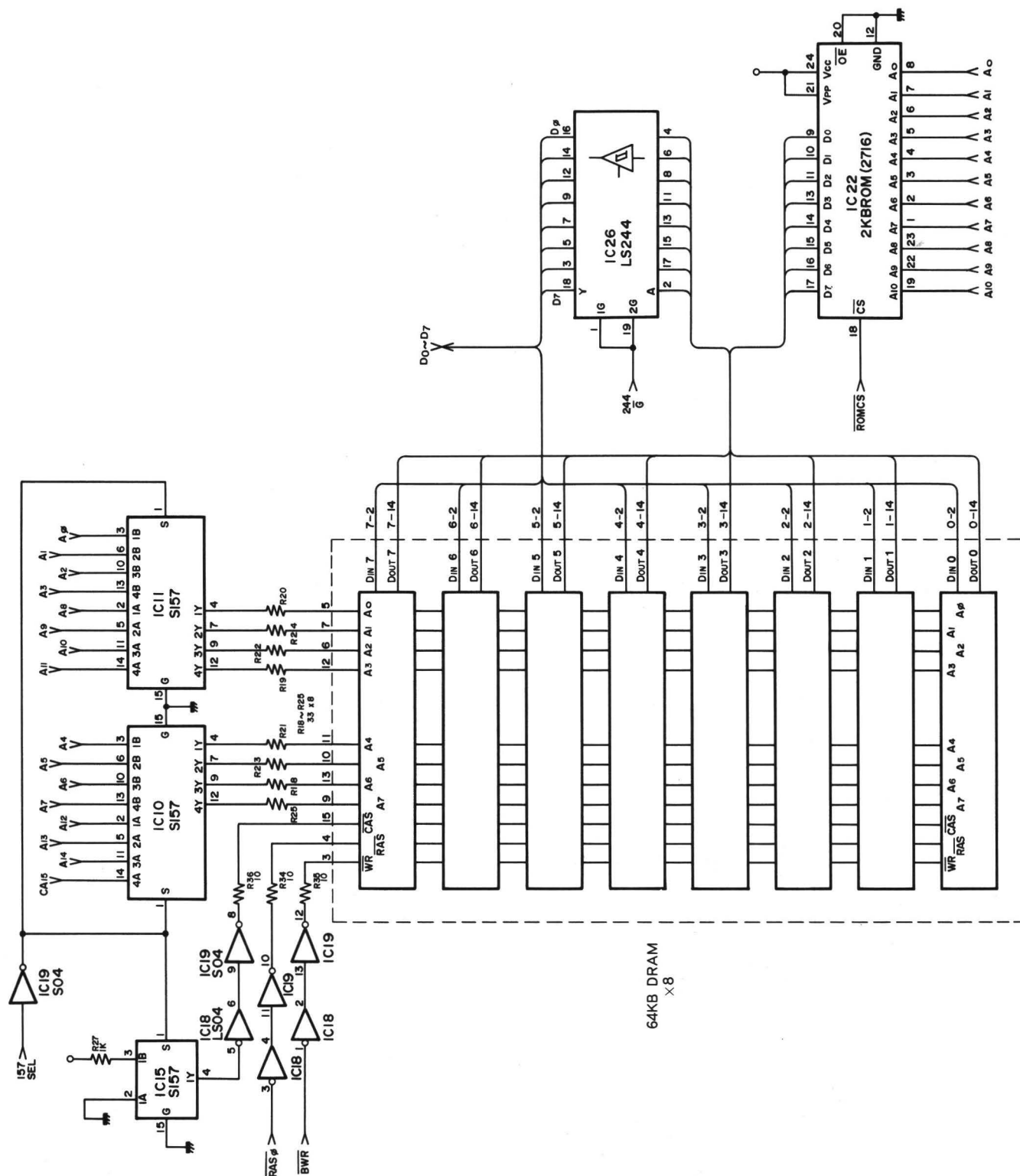



図 4-14 CPUボード〔4〕：RAMブロック

CN6 ──→ MZ-2000
CN2 ──→ MZ-1R01

信号	ピン	ピン	信号
GND	2	1	GND
GND	4	3	GND
GND	6	5	GND
+12V	8	7	+12V
+5V	10	9	+5V
+5V	12	11	+5V
−5V	14	13	−5V
D_2	16	15	D_3
D_1	18	17	D_4
D_0	20	19	D_5
D_7	22	21	D_6
a_1	24	23	a_0
a_3	26	25	a_2
a_5	28	27	a_4
a_7	30	29	a_6
a_9	32	31	a_8
a_B	34	33	a_A
a_D	36	35	a_C
$\overline{\mathrm{F4WR}}$	38	37	16M
$\overline{\mathrm{GOUT}}$	40	39	$\overline{\mathrm{Gse}\ell}$
Video QH	42	41	$\overline{\mathrm{QG}}$
$\overline{\mathrm{Hb}\ell\mathrm{an\ KF}}$	44	43	$\overline{\mathrm{Hsync}}$
$\overline{\mathrm{Vsync}}$	46	45	$\overline{\mathrm{Vb}\ell\mathrm{ank}}$
Vgate	48	47	Vid
$\overline{\mathrm{BMREQ}}$	50	49	bℓank
$\overline{\mathrm{BWR}}$	52	51	$\overline{\mathrm{BRFSH}}$
		53	$\overline{\mathrm{BRD}}$

図 4-15 CPUボード〔5〕：端子表

図 4-16 CRTディスプレイコントロール部

図 4-17 カセットテープデッキコントロール部

DBOXD0044PAZZ
-5V
+15V
+12V
GND
+5V
IC1 μPC7805H TA78005AP
IC41 μPC7815H TA78015AP
IC42 TL494CN
IC21 TL494CN
Q41 A770Y
Q42 A673C
Q21 A770Y
Q22 A673C
D41 S4VB10
D21 S4VB10
D42 CTB24
D22 CTB24
L41 Z0007
L21 Z0027
F41 T2.0A
F21 T2.0A
VR41 1K
VR21 1K
T101 P0079PA
T.F
L101 Z0005
C102 0.047/125V AC
C101 0.047/125V AC
F101 T1A
SW101
S0101

図 4-18 電源回路

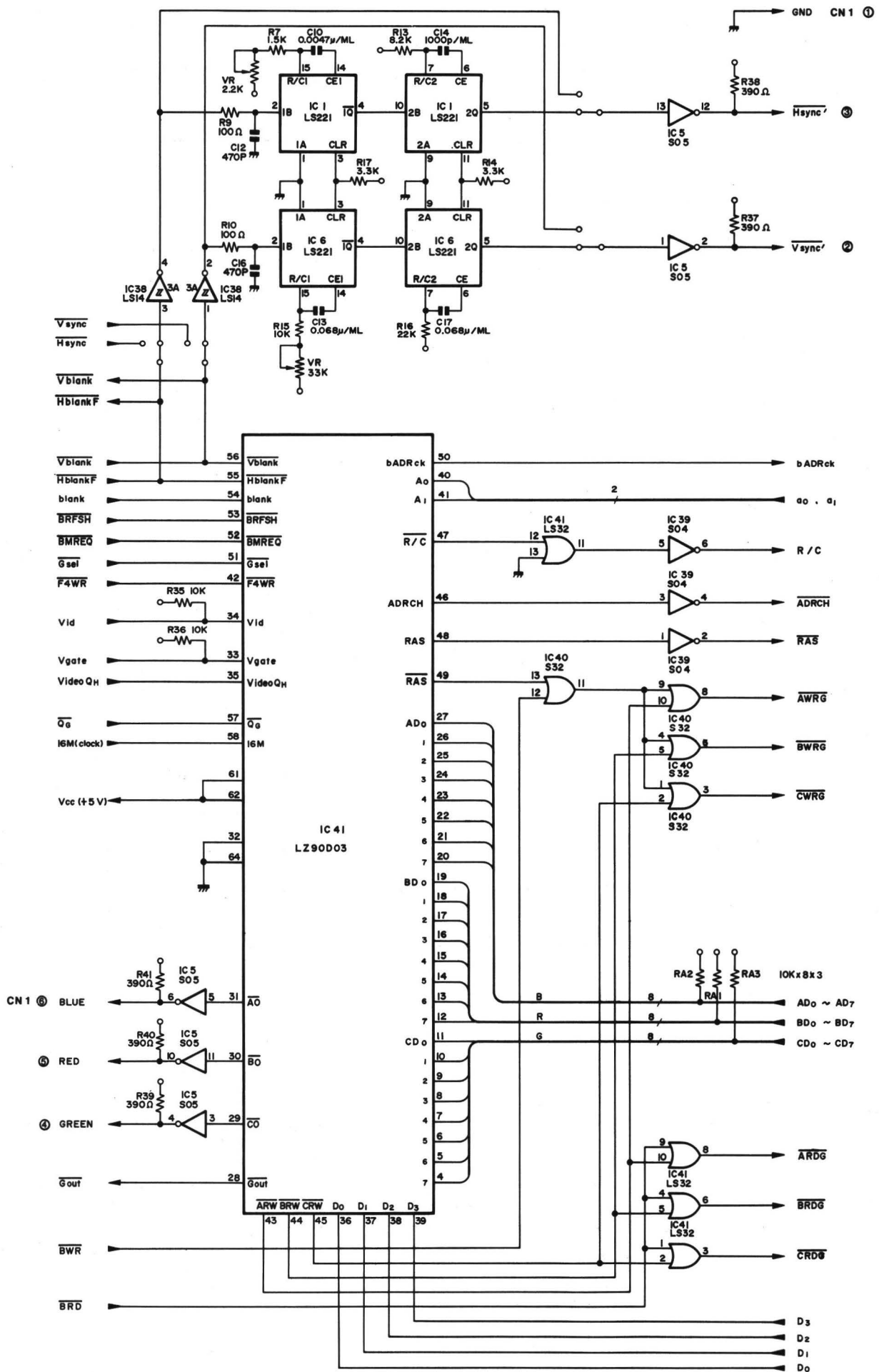



図 4-19 グラフィックボードMZ-1R01 コントロール部〔1〕……オプション

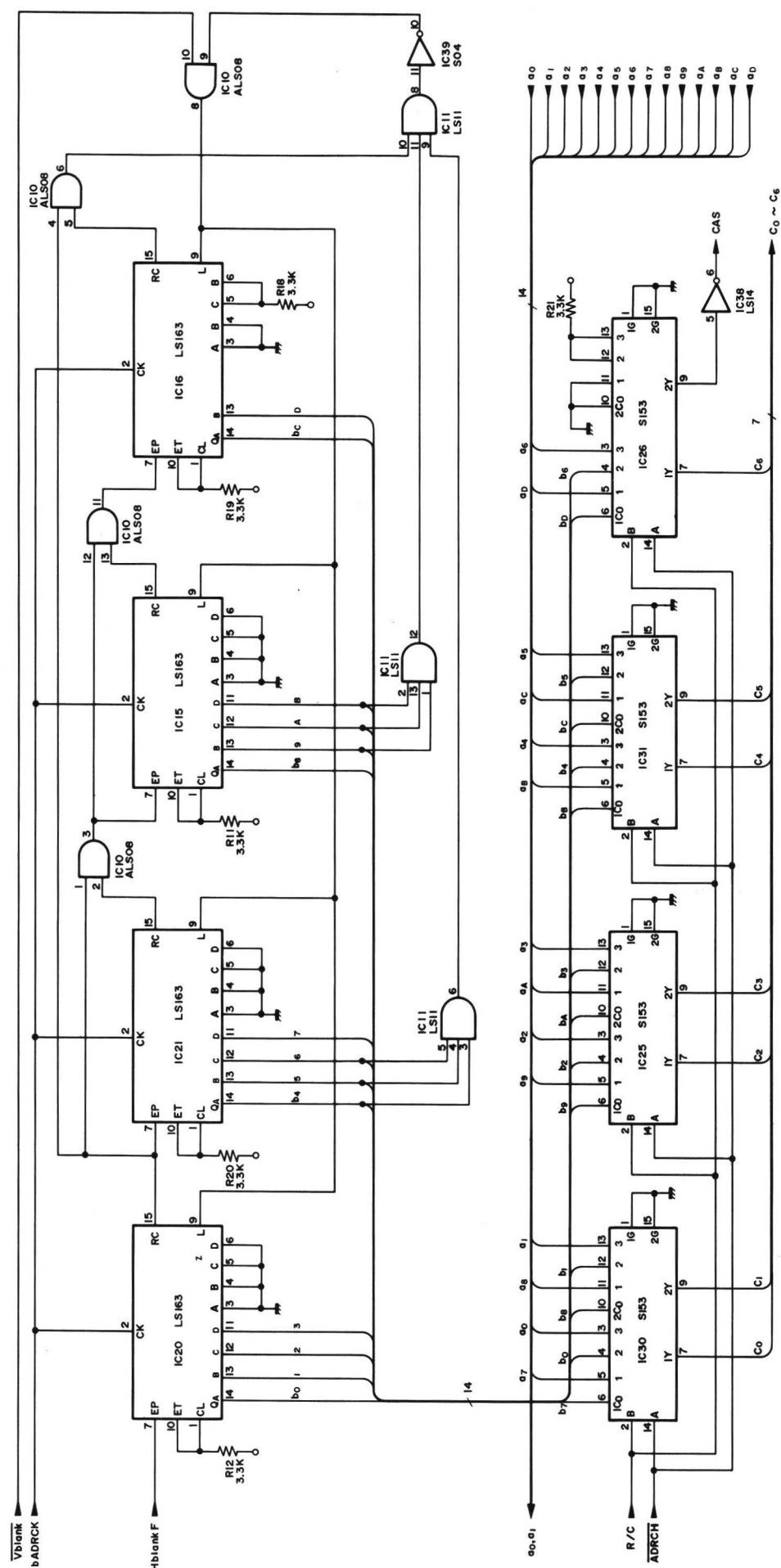



図　4-20　グラフィックボードMZ-1R01 コントロール部〔2〕……オプション

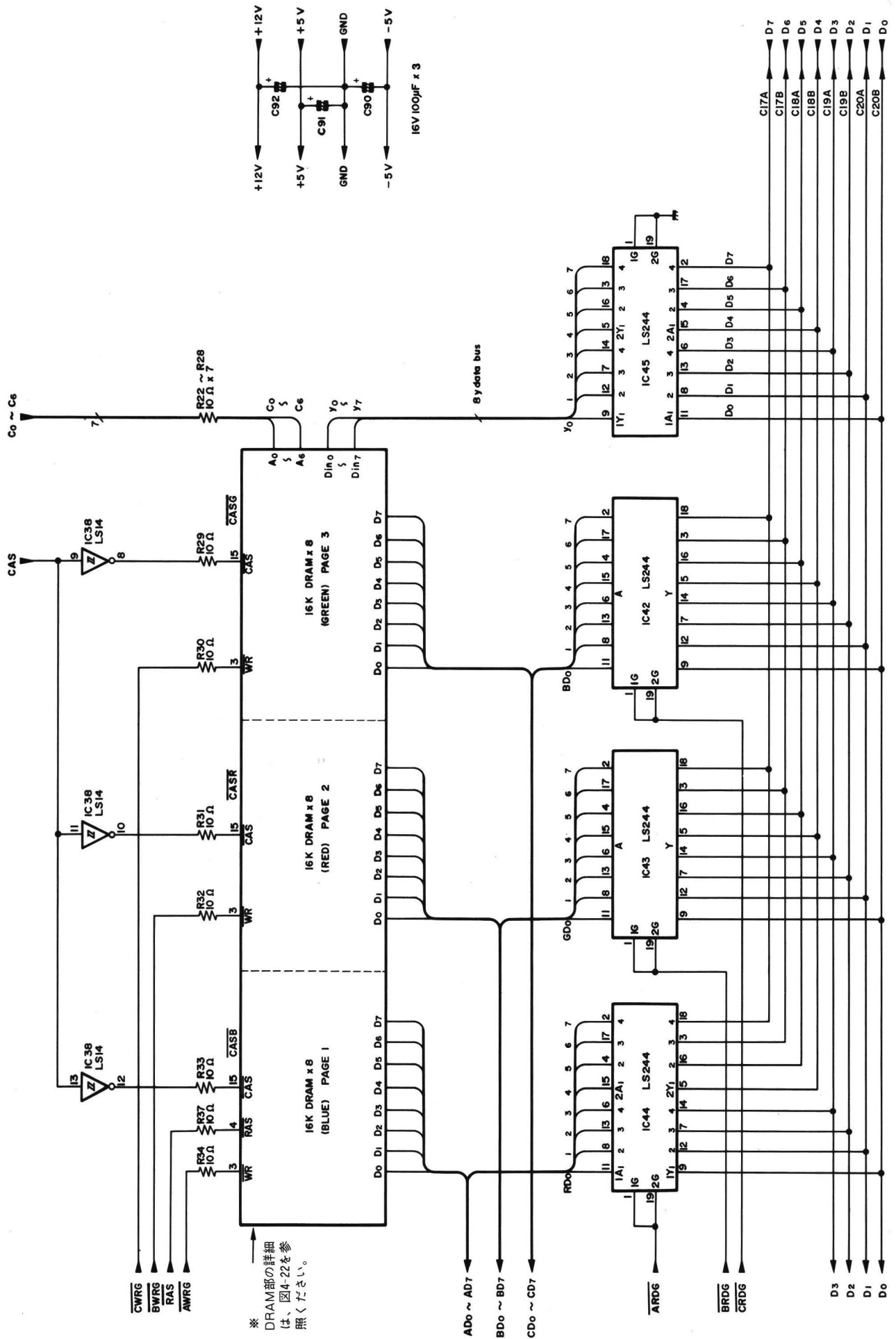



図 4-21 グラフィックボードMZ-1R01 RAMブロック〔1〕……オプション

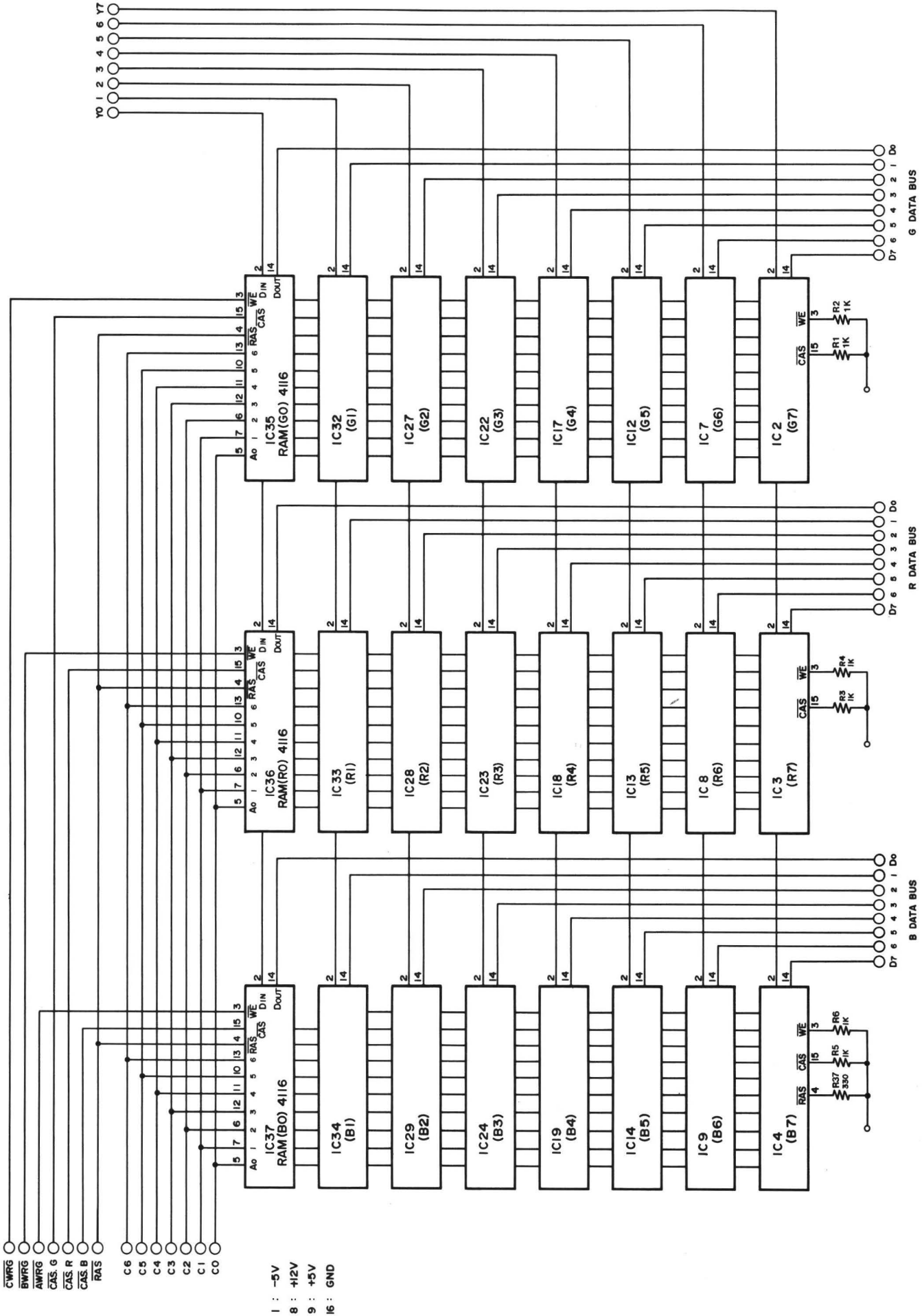



図 4-22 グラフィックボードMZ-1R01 RAMブロック詳細図

MZ-1U01　I/O MOTHER BOARD

CN5

A		B
D1	1	D0
D3	2	D2
GND	3	GND
D5	4	D4
D7	5	D6
GND	6	A0
RESET	7	A1
GND	8	A2
$\overline{\text{HALT}}$	9	A3
$\overline{\text{M1}}$	10	A4
GND	11	A5
$\overline{\text{WR}}$	12	A6
$\overline{\text{RD}}$	13	A7
GND	14	A8
$\overline{\text{IOREQ}}$	15	A9
$\overline{\text{MREQ}}$	16	A10
GND	17	A11
$\overline{\text{EXINT}}$	18	A12
IEO	19	A13
$\overline{\text{NMI}}$	20	A14
$\overline{\text{EXWALT}}$	21	A15
$\overline{\text{EXRESET}}$	22	ø

A: PARTS SIDE

+5V
RA1　1Kx5
IC1
LS08
IEO
IE11　IEO1
IE12　IEO2
IE13　IEO3
IE14　IEO4
CN1
CN2
CN3
CN4
$\overline{\text{INT1}}$
$\overline{\text{INT2}}$
$\overline{\text{INT3}}$
$\overline{\text{INT4}}$
EXINT

CN1~CN4

A		B
+5V	1	+5V
D2	2	D3
D1	3	D4
D0	4	D5
GND	5	D6
A15	6	D7
A14	7	ø
A13	8	$\overline{\text{M1}}$
A12	9	$\overline{\text{WR}}$
A11	10	$\overline{\text{RD}}$
A10	11	$\overline{\text{IOREQ}}$
A9	12	$\overline{\text{MREQ}}$
A8	13	GND
A7	14	$\overline{\text{HALT}}$
A6	15	IEI
A5	16	IEO
A4	17	RESET
A3	18	$\overline{\text{EXRESET}}$
A2	19	$\overline{\text{EXINT}}$
A1	20	$\overline{\text{EXWAIT}}$
A0	21	$\overline{\text{NMI}}$
GND	22	GND

A: PARTS SIDE

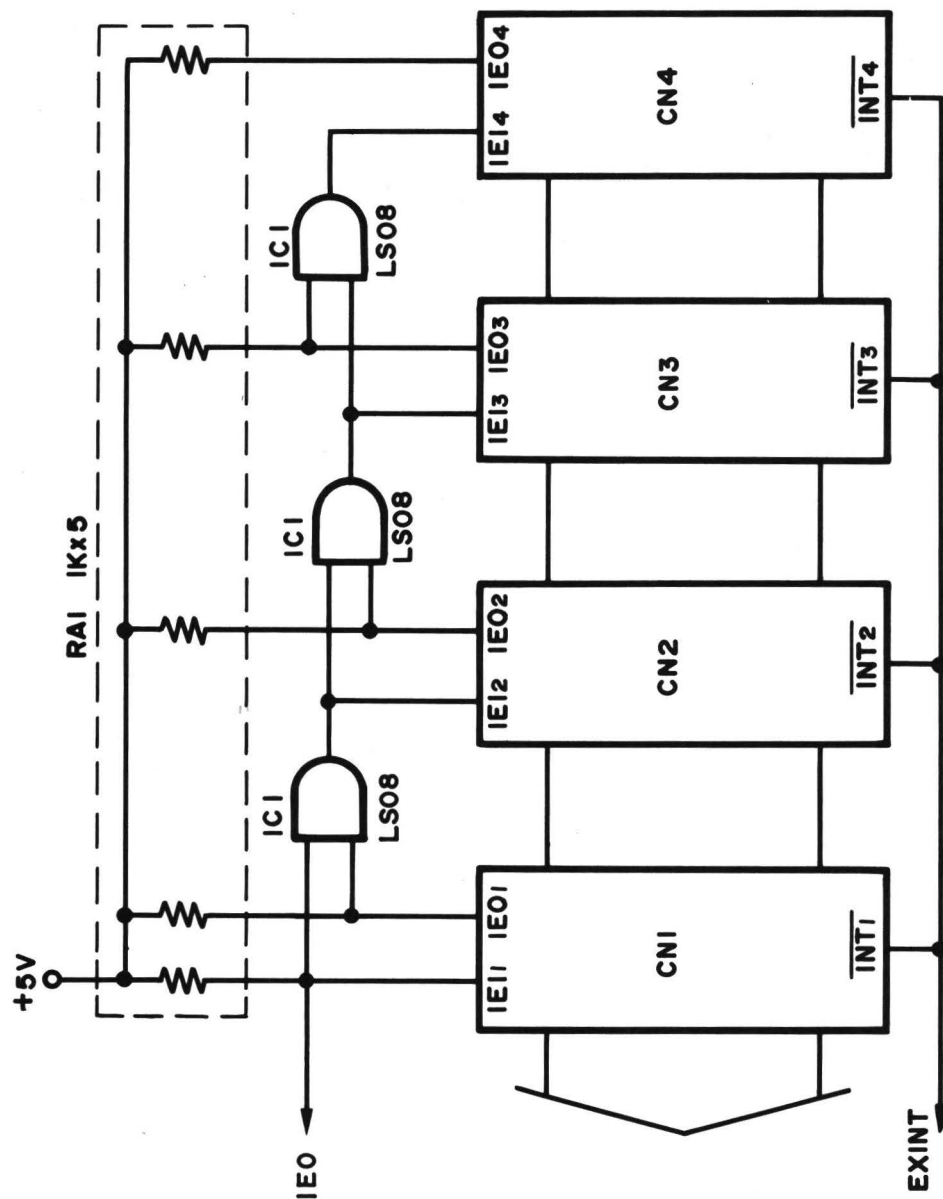



図 4-23　拡張I/OポートユニットMZ-1U01(オプション)

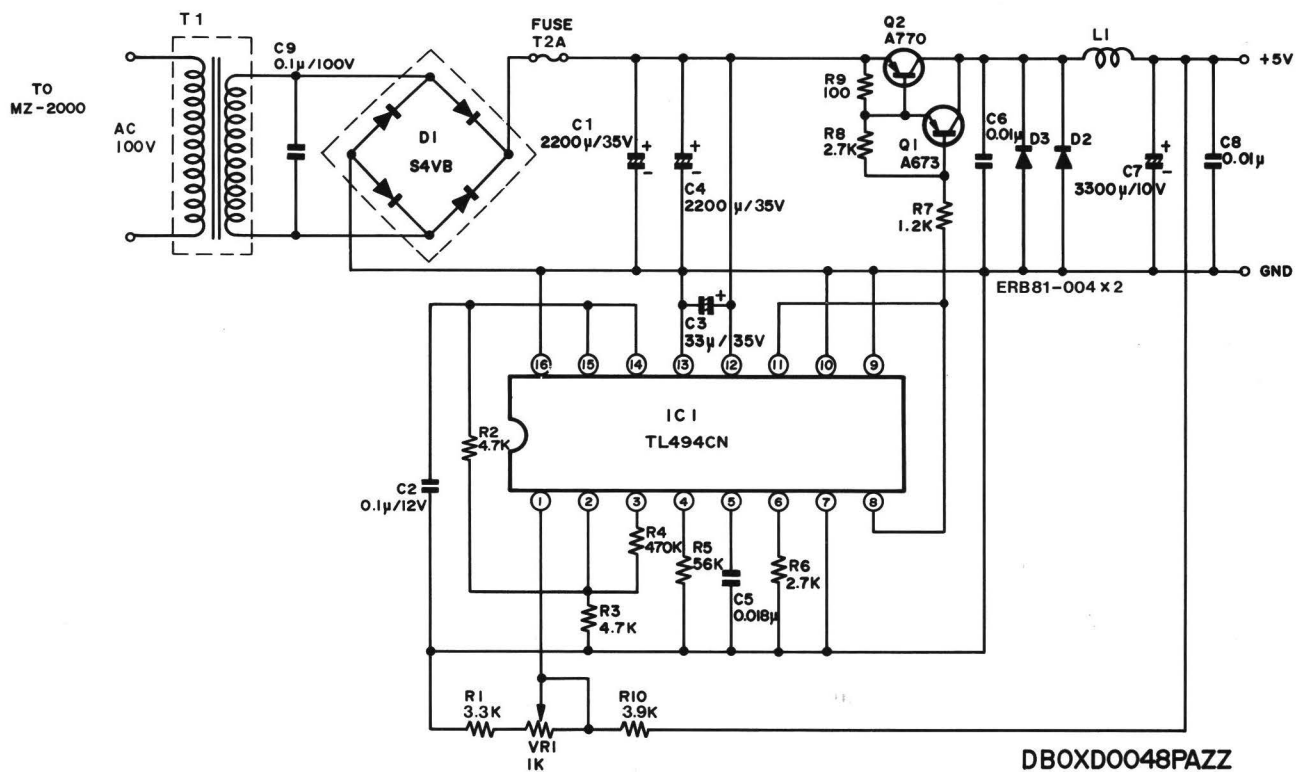



図 4-24 MZ-1U01電源回路(オプション)

付　　録　Appendix

付録には、LH0080A(Z80A-CPU)およびLH0081A(Z80A-PIO)の各テクニカルデータを参考のため掲載しています。(A.1とA.2)

A.3はMZ-2000の仕様の一覧

A.4には、MZ-2000の取り扱いの注意がまとめられています。

A.1 Z80A CPUテクニカルデータ

1. アーキテクチャ

Z－80A CPUの内部構成のブロック図を図１－１に示す。この図はCPU内部の主要な部分を示している。それぞれについては以下の説明を参照されたい。

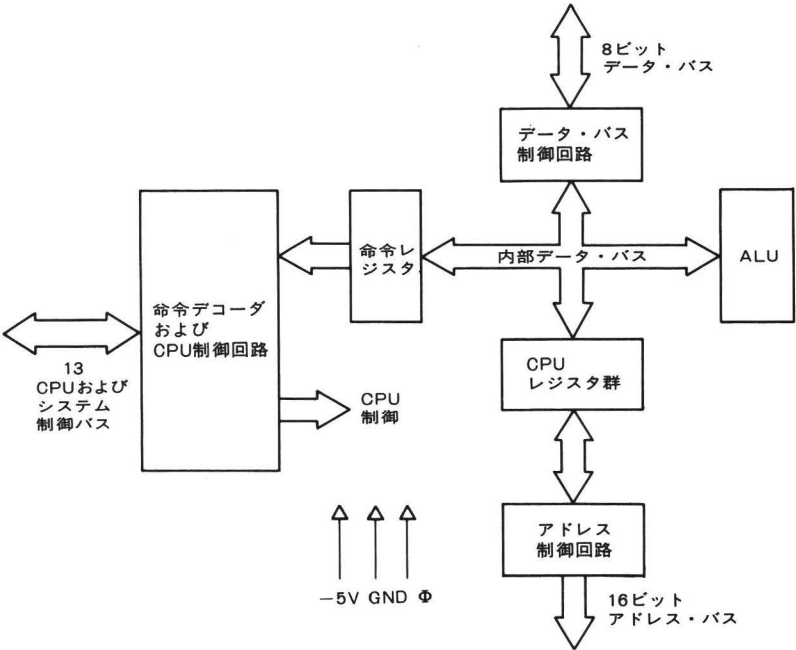



図1－1　Z－80A CPUの内部構成のブロック図

1.1 CPUレジスタ群

Z－80A CPUには207ビットの読み出し／書き込みメモリがあり、プログラマによって任意にアクセスできるようになっている。図１－２にこのメモリの配列を示す。これらは16個の８ビット・レジスタ、４個の16ビット・レジスタから成っている。

Z－80Aのレジスタはすべてスタティック RAMで構成されている。

６個の汎用レジスタがそれぞれ２セット用意されていて、これを単独の８ビット・レジスタとして使用してもよいし、対にして16ビット・レジスタとして用いることもできる。

２セットあるアキュムレータ、フラグ・レジスタについても同様である。

専用レジスタ

1．プログラム・カウンタ（PC）

プログラム・カウンタは現在実行中の命令のメモリ・アドレス16ビットを保持しており、CPUはPCで示されるメモリ・アドレスから命令をフェッチする。

このPCはその内容をアドレス線に送り出すと、インクリメンタによりPCの値を自動的に＋１する。プログラム・ジャンプの場合、インクリメンタは動作せず、新しい値が直接PCにセットされる。

2．スタック・ポインタ（SP）

スタック・ポインタは外部RAM上のスタック領域のその時点での最上位のアドレス16ビットを保持するものである。外部スタックはLIFO（Last-in, First-out）ファイルとして構成される。

データはPUSHおよびPOP命令により、CPUの指定レジスタからスタックへ、またスタックからCPUの指定レジスタへ転送される。

データのスタックからの取り出し（POP）は一番最後に押し込んだ（PUSH）データから行われる。このスタックは、多重レベルの割り込み、無限のサブルーチン・ネスティング、あるいは種々の形のデータ操作などを簡単にする場合に有効である。

主レジスタ・セット		補助レジスタ・セット		
アキュムレータ A	フラグ F	アキュムレータ A′	フラグ F′	
B	C	B′	C′	汎用レジスタ群
D	E	D′	E′	
H	L	H′	L′	

割り込みベストル I	メモリ・リフレッシュ R	
インデックス・レジスタ IX		専用レジスタ群
インデックス・レジスタ IY		
スタック・ポインタ　SP		
プログラム・カウンタ　PC		

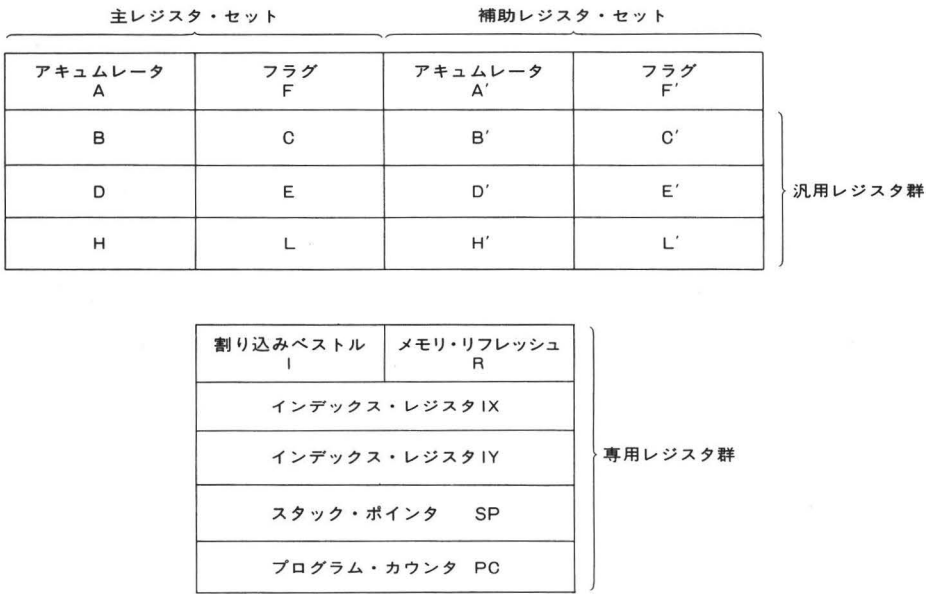



図1－2　Z－80A CPUのレジスタ構成

3．インデックス・レジスタ（IX&IY）

インデックス・モードのアドレシング用として16ビットの基準アドレスを保持する2個の独立したインデックス・レジスタIX、IYがある。

このモードでは、インデックス・レジスタはデータを出し入れするメモリ領域を指定するための基準アドレスとして使用される。

インデックス・アドレシング命令では、このレジスタの内容に1バイトのディスプレイスメント(偏位値)を加算した値が実効アドレスとなる。この偏位値は2の補数の符号付整数で与えられる。

このアドレシング・モードは種々のプログラム、とくに、データ・テーブルを参照するといったプログラムなどによく使用される。

4．割り込みページ・アドレス・レジスタ（I）

Z－80A CPUには、割り込みに応じてどのメモリの位置へでも、イン・ダイレクト・コール（間接サブルーチン・ジャンプ）できるモードがある。

この目的のために、Iレジスタが用意されていて、間接アドレスの上位8ビットにこのレジスタの内容が相当する。下位8ビットには、割り込みをかけてきたデバイスに対応するアドレスが当てられる。この方式によれば、割り込み処理ルーチンを動的にメモリのどこへでも配置でき、極めて短いアクセス時間で処理ルーチンへ飛ばせ得るという特長をもっている。

5．メモリ・リフレッシュ・レジスタ（R）

Z－80A CPUはメモリ・リフレッシュ・カウンタを内蔵しているので、スタティック・メモリと同じ手軽さでダイナミック・メモリを使用できる。

この　ビットのRレジスタは各命令のフェッチごとに自動的にインクリメントされる。

Rカウンタのデータは CPUがフェッチした命令をデコードし、実行している間に、リフレッシュ制御信号と同期してアドレス・バスの下位に乗せられる。

このリフレッシュのモードは、プログラマがとくに気をつかう必要もなく、またCPUの動作を遅らせるものでもないという特長がある。Rレジスタにテストの目的でプログラムによってデータをロードすることもできるが、普通は使用しない方がよい。

リフレッシュの期間、アドレス・バスの上位8ビットにはIレジスタの内容が出力する。

アキュムレータとフラグ・レジスタ

CPUは2組の独立した8ビットのアキュムレータと、それと組み合わさった2組の8ビットのフラグ・レジスタをもっている。

アキュムレータは8ビットの算術、論理演算の結果を保持する。一方、フラグ・レジスタは8ビットあるいは16ビットの演算結果の状態、たとえば結果が零に等しいか否かをセットする。

プログラマによって1つの交換命令を用いて、アキュムレータとフラグの対(ペア)のうちどちらか使いやすい方を選べばよい。

汎用レジスタ

2組の対になった汎用レジスタ群があり、それぞれ単独で8ビットのレジスタとして使用でき、また16ビットのレジスタ・ペアとしても使用できる。一方のセットとしてBC、DE、HLがあり、他方、BC′、DE′、HL′のセットがある。

プログラムによって、どの時点においても交換命令を用いてどちらかの側のレジスタ群を作業用として使用することができる。システム内で高速の割り込み応答が要求される場合、この手法を使ってアキュムレータ、フラグ類、汎用レジスタ群の内容を他方へすばやく退避させてもよいだろう。

1つの簡単な交換命令を用いるだけでルーチン間の移行が行える。

これにより割り込み、あるいはサブルーチン処理の期間、レジスタの内容を外部スタックへ移したり、戻したりする必要がなくなるので、割り込みサービス時間を大幅に短縮するのに役立つ。

これらの汎用レジスタ群は、どのような幅広い用途に対しても利用し得るものである。

単純なプログラム、とくにROMベースのシステムなどで、簡単な読み出し/書き込みメモリが必要な場合に、汎用レジスタで代用すればよい。

1.2 算術、論理演算ユニット(ALU)

8ビットの算術、論理演算命令はCPU内のALUで実行される。

ALUは各レジスタと内部バスとを接続しており、それらの間でデータの送受を行う。

ALUに関連した機能には次のようなものがある。

Add	(加　算)
Subtract	(減　算)
AND	(論理積)
OR	(論理和)
XOR	(排他的論理和)
Compare	(比　較)
Shift Left, Right	(左、右シフト)
Rotates arithmetic, logical	(算術的、論理的ローテイト)
Increment	(インクリメント；+1)
Decrement	(デクリメント；−1)
Set bit	(ビット・セット)
Reset bit	(ビット・リセット)
Test bit	(ビット・テスト)

1.3 命令レジスタ、CPU制御

各命令はメモリから読み出されてきて、命令レジスタに保持され、デコードされる。制御部はこの制御を行うとともに、レジスタ群からのまたはそれらへの、データの読み込み、書き出しに必要な制御信号を発生している。

さらに、ALUの制御信号や必要な外部制御信号を作り出す。

2. 端子説明

Z－80A CPUは標準型40ピンDIPパッケージに納められている。入出力端子配置図を図2－1に示す。以下にそれぞれの機能を説明する。

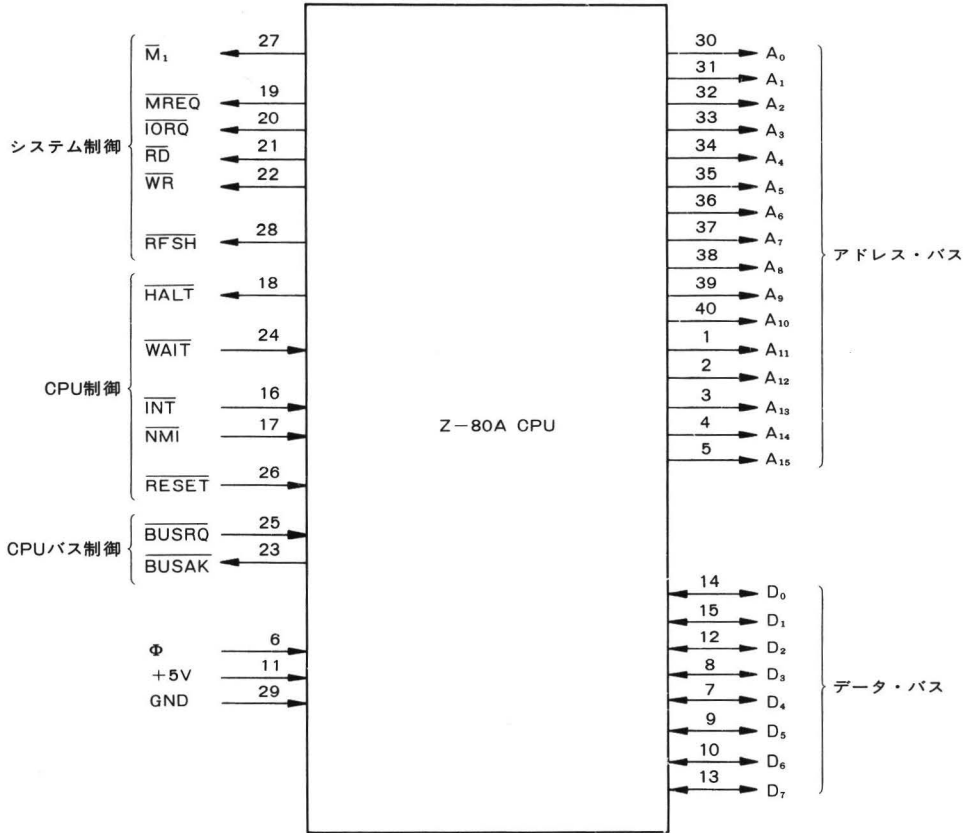



図2－1　Z－80A CPUの端子配置図

A_0－A_{15}
（アドレス・バス）

トライ・ステート、アクティブ“H”。
A_0－A_{15}は16ビットのアドレス・バスである。このアドレス・バスにより、メモリ（最大64Kバイト）内のデータや入出力デバイスのデータの送受のためのアドレス指定を行う。入出力アドレシング用として下位8ビットを用いる。ユーザは直接256の入力ポートまたは256の出力ポートを選択できるようになっている。A_0がLSB（最下位ビット）となる。リフレッシュ期間には、アドレス・バスの下位7ビットにリフレッシュ用の実効アドレスが乗せられる。

D_0－D_7
（データ・バス）

トライ・ステート入出力、アクティブ“H”。
D_0－D_7は8ビットの双方向性データ・バスである。データ・バスはメモリ、入出力間、あるいはそれらの相互間で、データ交換を行うために用いる。

$\overline{M_1}$
（マシン・サイクル1）

出力、アクティブ“L”。
$\overline{M_1}$は、現在のマシン・サイクルが、命令実行中のOPコードのフェッチ・サイクルであるときに出力する。2バイトのOPコードの実行時には、$\overline{M_1}$はOPコードのフェッチ・サイクルごとに出されることに注意されたい。これらの2バイトのOPコードを持つ命令では、先頭のOPコードはCB_H、DD_H、ED_HあるいはFD_Hで始まっている。
さらに、M_1は割り込みアクノリッジ時にも出力する。CPUは$\overline{M_1}$と$\overline{IORQ}$により外部のデバイスに対して割り込みアクノリッジを通知する。

$\overline{MREQ}$
（メモリ要求）

トライ・ステート出力、アクティブ“L”。
メモリ要求信号は、メモリ読み出し、あるいはメモリ書き込みのための実効アドレスが、アドレス・バスに乗っているときに出される。

$\overline{IORQ}$
（入出力要求）

トライ・ステート出力、アクティブ〝L″。
$\overline{IORQ}$信号は入出力の読み出し、書き込みのための実効入出力アドレスが、アドレス・バスの下位8ビット上に乗っているときに出される。また$\overline{IORQ}$信号は割り込みアクノリッジ時に$\overline{M_1}$とともに出力され、この2つの信号により割り込み応答ベクトルをデータ・バス上に乗せてもよいことを入出力装置に知らせる。このM_1の期間では入出力装置の処理は行われず、割り込みアクノリッジの処理が行われる。

$\overline{RD}$
（メモリ読み出し）

トライ・ステート出力、アクティブ〝L″。
$\overline{RD}$はCPUがメモリ、あるいは入出力デバイスからデータを受け入れられる期間で出される。指定された入出力デバイス、あるいはメモリのデータは、この信号でゲートして、CPUデータ・バスに乗せるとよい。

$\overline{WR}$
（メモリ書き込み）

トライ・ステート出力、アクティブ〝L″。
$\overline{WR}$は、CPUデータ・バスに、指定したメモリあるいは入出力デバイスにストアするべきデータが乗っているときに出される。

$\overline{RFSH}$
（リフレッシュ）

出力、アクティブ〝L″。
$\overline{RFSH}$は、ダイナミック・メモリのためのリフレッシュ・アドレスがアドレス・バスの下位7ビットに乗っているときに出される。なお、ダイナミック・メモリをリフレッシュ(リフレッシュ読み出し)する場合は$\overline{MREQ}$信号も必要である。

$\overline{HALT}$
（ホールト）

出力、アクティブ〝L″。
$\overline{HALT}$は、CPUがHALT命令を実行し、ノン・マスカブルあるいは、マスカブルな割り込み待ちとなったときに出される。
ホールト時には、CPUはNOP命令を実行することによりリフレッシュ信号を出し続けている。

$\overline{WAIT}$
（ウエイト）

入力、アクティブ〝L″。
この入力信号を用いて、メモリあるいは入出力デバイスがデータの送出の用意ができていない旨をZ－80A CPUに伝える。この信号がアクティブである限り、CPUはウェイト状態を続ける。この期間リフレッシュ信号は出力しないので注意されたい。この信号を用いることによりどのような動作速度のメモリや入出力デバイスに対してもCPUを同期させることができる。

$\overline{INT}$
（割り込み要求）

入力、アクティブ〝L″。
割り込み要求信号は入出力デバイスから発せられる。ソフト的に(プログラムで)、割り込みの許可フラグ（IFF）がセットしてあり、$\overline{BUSRQ}$信号がノン・アクティブならば、割り込み要求は実行中の命令が終わり次第受け付けられる。
CPUは割り込みを受け付ければ、アクノリッジ信号（M_1期間の$\overline{IORQ}$）を次の命令サイクルの始めで送出する。第5章（CPUを制御する命令群）に詳細を示すがマスク可能な割り込みは3種ある。

$\overline{NMI}$
（ノン・マスカブル割り込み）

入力、立ち下がりエッジ検知。
ノン・マスカブル割り込み要求は$\overline{INT}$より高位の優先順位を持っていて、現在実行中の命令の最後のTサイクルの立ち上がりまでに入力していると、その命令完了後受け付けられる。これは割り込みの許可フラグの状態には関係しない。$\overline{NMI}$入力でCPUは自動的に、0066_Hの番地からリスタートする。プログラム・カウンタの内容は、割り込みがかけられた元のプログラムへ戻れるように、外部スタックへ自動的に退避される。
連続してWAITサイクルがあれば、NMIは待たされ、また$\overline{BUSRQ}$は$\overline{NMI}$より優先順位が高い。

$\overline{RESET}$

入力、アクティブ〝L″。
$\overline{RESET}$入力によりプログラム・カウンタは零となり、CPUは初期化される。このとき次の状態となる。

1）割り込みの許可フラグがリセットされる。
2）レジスタI＝00_Hにセットされる。
3）レジスタR＝00_Hにセットされる。
4）割り込みはモード0にセットされる。

リセット期間中、アドレス・バスとデータ・バスは高インピーダンスとなり、すべての制御出力は非アクティブ状態となる。

$\overline{\text{BUSRQ}}$
(バス要求)

入力、アクティブ〝L″。
バス要求信号により、CPUのアドレス・バス、データ・バスおよびトライ・ステート出力の制御線は、他のデバイスがバスを使用できるようにするため高インピーダンスとなる。
$\overline{\text{BUSRQ}}$がアクティブになったとき、その時点でのCPUのマシン・サイクルが終わった瞬間に、バス線は高インピーダンスとなる。

$\overline{\text{BUSAK}}$
(バス・アクノリッジ)

出力、アクティブ〝L″。
バス・アクノリッジは、CPUのアドレス・バス、データ・バスおよびトライ・ステート制御バスが高インピーダンスとなり、外部デバイスがこれらのバスを使用できるようになった時点で、要求のあったデバイスに対して出力する。

Φ

単相のTTLレベルのクロック入力。

3. タイミング

Z－80A CPUは基本的な操作を組み合わせ、1ステップずつ命令を実行していく。その操作は次のものから構成される。

メモリ・読み出し、書き込み、

入出力デバイス・読み出し、書き込み

割り込み、アクノリッジ

すべての命令はたんにこれらの基本的操作を組み合わせたものである。この基本操作は3～6クロックで行われる。これらの操作はCPUを外部デバイスの速度に同期させるために延長することもできる。基本のクロック期間をTサイクルとし、基本操作をMサイクル（マシン・サイクル）とする。図4－1に、ある命令に対して特定のMとTサイクルの配列がどのようになっているかを例として示す。ただし、この例では、3マシン・サイクル（M1、M2、M3）の命令について示してある。

最初のMサイクルはどの命令でも命令のフェッチ・サイクルである。このフェッチ・サイクルには、4、5、6Tサイクルの3種があるが、ウェイト信号（次節で詳述する。）を用いると、このサイクル数を変えることもできる。このフェッチ・サイクル(M1)は、次に実行すべき命令のOPコードをフェッチするための期間である。

後続するMサイクルで、CPUとメモリ、入出力デバイス間のデータ転送が行われる。これらも基本的には3～5Tサイクルであるが、外部デバイスと同期させるために入力するWAIT信号によりサイクル数は変化する。

次項から基本マシン・サイクル内でのタイミングについて説明する。

各命令の詳細なタイミングを第10章に示す。

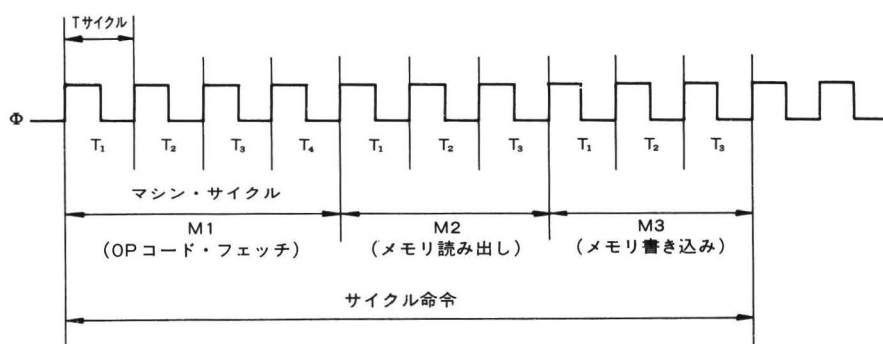



図3－1　　CPU基本タイミング例

すべてのCPUタイミングは、図3－2に示すような、いくつかの非常に簡単なタイミング図に分解できる。ウェイトを含む場合と含まない場合の基本動作タイミングを以下において示す。(ウェイト状態は遅いメモリや入出力デバイスにCPUを同期させるために付加される。)

3.1 命令フェッチ

図3－2にM1サイクル（OPコード・フェッチ）のタイミングを示す。プログラム・カウンタの内容はM1サイクルの先頭でアドレス・バス上に乗せられ、クロックの半サイクル後MREQがアクティブとなる。このときすでに、メモリに対するアドレス信号は安定しているので、MREQの立ち下がりエッジをダイナミック・メモリへのチップ・イネーブル・クロックとして直接使用することができる。

RDもメモリから読み出すデータをCPUのデータ・バスに乗せてもよいことを指示するためにアクティブとなる。

CPUはデータ・バス上にあるメモリからのデータを、T_3のクロックの立ち上がりエッジでサンプルする。その立ち上がりエッジで、RD、MREQ信号が切り換わるが、CPUはデータのサンプリングを、RDが非アクティブになる前に行っている。

フェッチ・サイクルのクロックT_3、T_4のステートはダイナミック・メモリをリフレッシュするために使用する。（この期間においては、CPU内ではフェッチされた命令のデコードとその実行を行っているので、外部に対する他の動作は行われない。）

T_3、T_4の期間で、アドレス・バスの下位7ビットにメモリのリフレッシュ・アドレスが乗せられ、RFSH信号がアクティブとなる。この信号により、すべてのダイナミック・メモリのリフレッシュがこの期間内に行われるように指示する。

注意する点として、データ・バスに接続されているはずの他のメモリ・セグメントからデータが乗らないように、リフレッシュの期間にはRD信号は出されていない、ということがあげられる。

リフレッシュ期間のMREQは、すべてのダイナミック・メモリに対するリフレッシュのための読み出しを行うために用いられる。リフレッシュ信号は単独では使用できない。その理由はリフレッシュ・アドレスは、MREQが出力している期間においてのみ安定しているので、MREQと併用しなければならないからである。

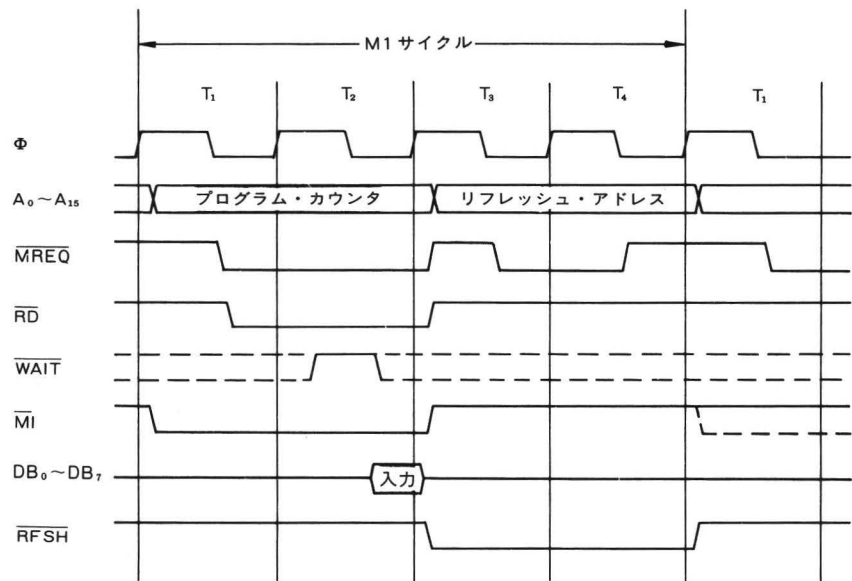



図3－2　命令OPコード・フェッチ(M1サイクル)

図３－２AにWAIT信号が入っている場合のフェッチ・サイクルの遅延状態を示す。T_2およびそれに続く各T_wのクロックΦの下がりエッジにおいて、CPUはWAIT信号をサンプルする。このサンプル時点において、もしWAITがアクティブならば、次のサイクルはさらに待ち状態になる。この方法を用いると、どのようなタイプのメモリ・デバイスのアクセス時間に対しても読み出し時間を延長することによって対処できる。

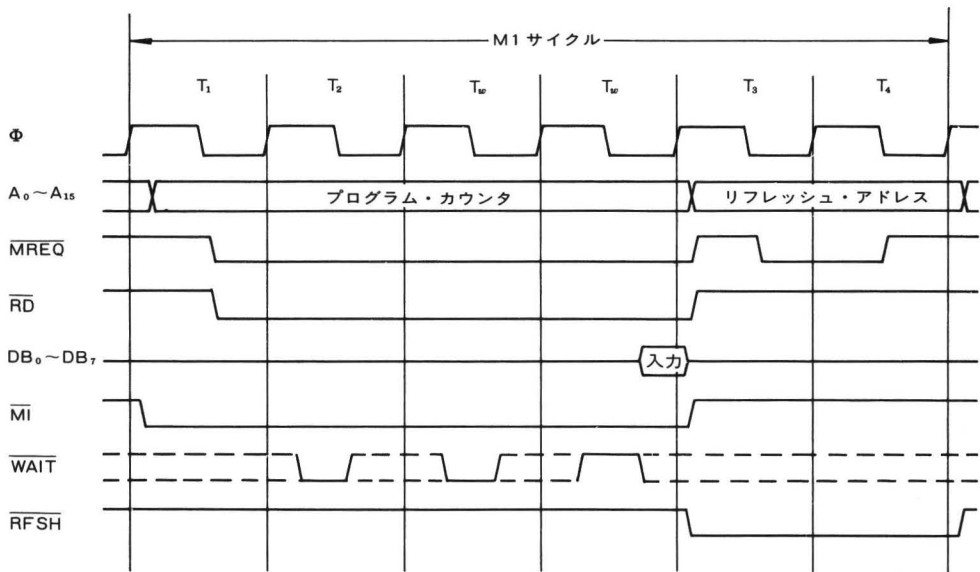



図3－2A　待ち状態を含む命令OPコード・フェッチ

3.2 メモリ読み出し、書き込み

図3－3はOPコード・フェッチ（M1サイクル）以外の場合のメモリの読み出し、書き込みサイクルのタイミング図である。

これらのサイクルは、メモリ側から出されるWAIT信号がなければ、通常3クロック・サイクルの長さである。

MREQ信号とRD信号はフェッチ・サイクルの場合と同様に用いられる。

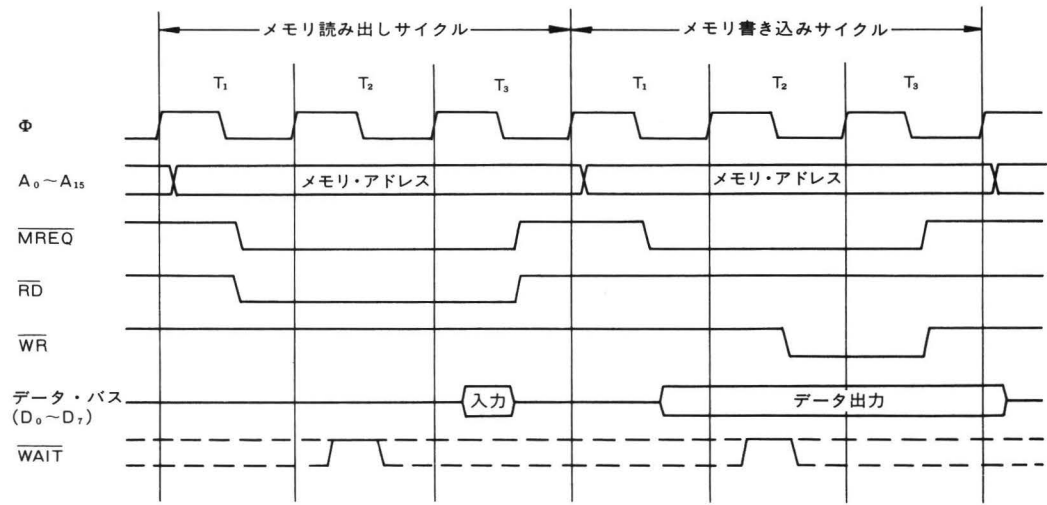



図3－3　メモリ読み出し、書き込みサイクル

メモリ書き込みサイクルにおいても、MREQはアドレス・バスが安定したときにアクティブになるので、これをダイナミック・メモリのチップ・イネーブル用に直接使用することができる。

WR出力はデータ・バスが安定になったときアクティブとなるので、これを直接メモリの読み出し、書き込みパルスとして使用してもよい。実際上、このWR出力はほとんどのタイプの半導体メモリの読み出し、書き込み信号として使用できる。

さらに、WR信号はアドレス・バス、データ・バスの内容が変化するサイクルの前のTサイクルの中央で非アクティブとなるので、実際上どんなタイプの半導体メモリのオーバ・ラップ条件に対しても、問題なく使用できる。

図3－3Aにメモリの読み出しや書き込み動作時にWAIT要求信号がある場合の状態を示す。

この動作はさきにフェッチ・サイクルのところで述べた動作と同様である。実際には、メモリの読み出しと書き込みサイクルは同時にはおこり得ないが、この図では便宜上これらを同一図上に書いてある。

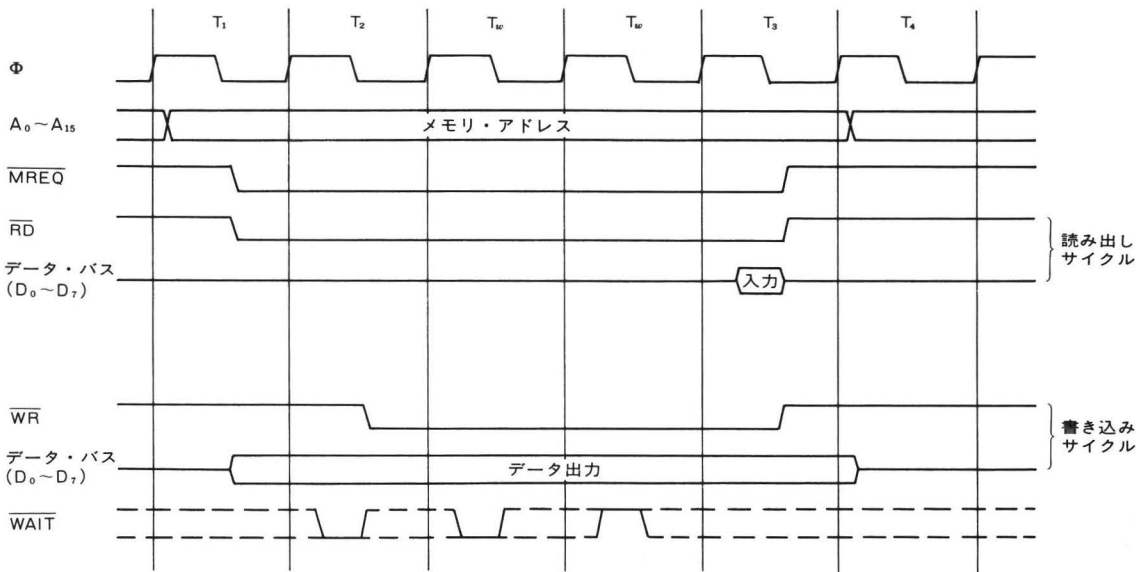



図3－3A　待ち状態を含むメモリ読み出し、書き込みサイクル

3.3 入力、出力サイクル

図3－4に入出力に対する読み出しや書き込みの動作を示す。この場合、自動的に1つの待ち状態T_Wサイクルが挿入される点に注意されたい。

このようにした理由は入出力装置の処理を行う場合、IORQがアクティブになってからCPUがWAIT信号をサンプルするまでの時間が非常に短いため、もしこの余分なステートT_Wがなければポート・アドレスをデコードしてからWAIT信号を起動していたのでは時間が足りないからである。

さらに、この待ち状態を設けなければ、MOS入出力デバイスを組み入れて、CPUを最高速で動作させるということもむずかしくなる。

このT_Wの期間にWAIT要求信号をサンプリングする。メモリの読み出しの場合と同様に、入出力読み出し動作においても指定したポートをデータ・バスに接続するためにはRD信号を使用すればよい。入出力書き込み動作において、入出力ポートのクロックとしてWR信号を用いるが、充分なオーバ・ラップ時間が自動的に作られるのでこの立ち上がりエッジをデータ・クロックとして使用することができる。

図3－4AにWAIT信号が入った場合に、余分の待ち状態サイクルが追加されるようすを示す。この動作はさきに述べた例（図3－2、図3－2Aなど）と同様である。

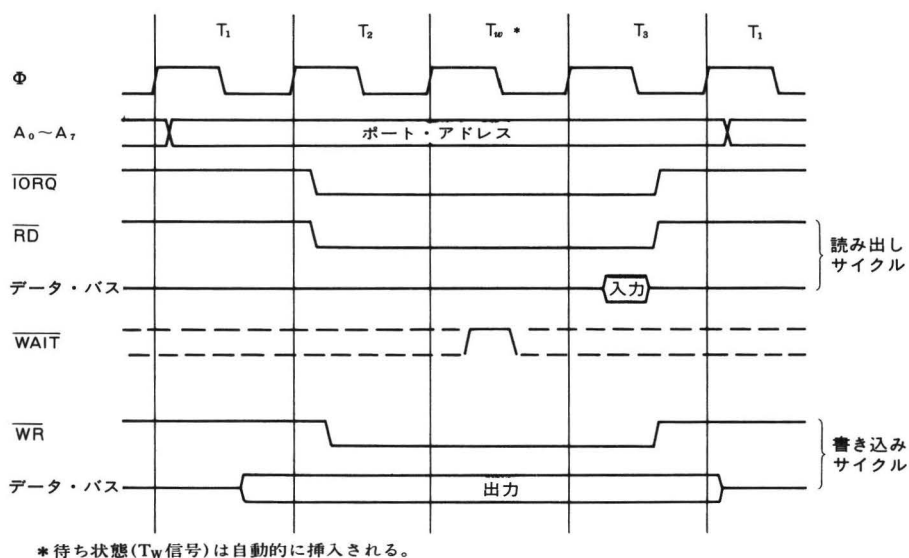



図3－4　入力、出力サイクル

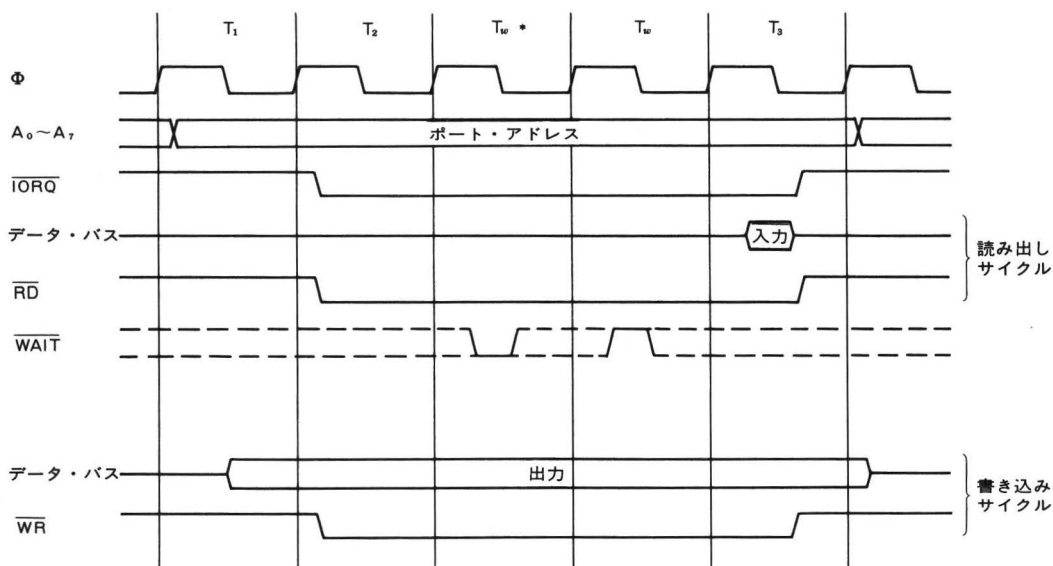



図3－4A　待ち状態を含む入力、出力サイクル

3.4 バス要求／アクノリッジ・サイクル

図３－５にバス要求／アクノリッジ・サイクルのタイミングを示す。

CPUは、各マシン・サイクルの最終クロックの立ち上がりエッジでBUSRQ信号を調べ、もしBUSRQ信号がアクティブならば、次のクロック・パルスの立ち上がりエッジでアドレス、データ、トライ・ステート制御線を高インピーダンスにする。この時点以降、外部のデバイスによりこれらのバスを制御でき、メモリと入出力間のデータ転送に使用することが可能になる。(これは一般にサイクル・スティール方式のダイレクト・メモリ・アクセスDMAとして知られている。)

バス要求に対するCPUの応答遅れは大きくても1マシン・サイクル長であり、外部のコントローラは必要な時間だけバスを制御できる。

しかしダイナミック・メモリを使用していて、長時間DMAサイクルを続けたい場合には、外部の制御回路によりダイナミック・メモリのリフレッシュが行えるようにしておかなければならない。

このような状態は、多数のデータ・ブロックをDMAで転送する場合にのみ生じる。

さらに、バス利用可の状態では、NMI信号あるいはINT信号による割り込みは無視される点に注意する必要がある。

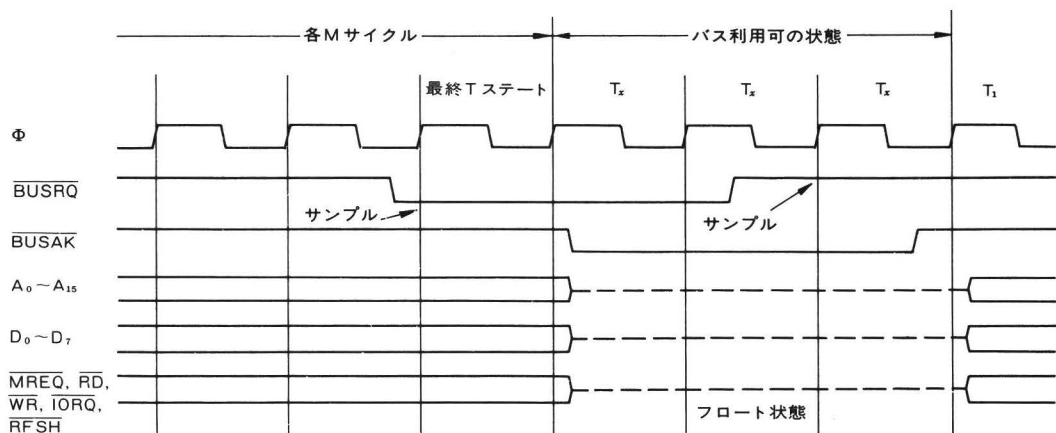



図3－5　バス要求／アクノリッジ・サイクル

3.5 割り込み要求／アクノリッジ・サイクル

図３－６に、割り込みサイクルに関連したタイミング例を示す。

CPUは各命令実行の最終クロックの立ち上がりエッジで割り込み信号INTをサンプルする。

CPU内部の割り込みイネーブル・フリップ・フロップがセットされていなければこの信号は受け付けられない。また、BUSRQ信号がアクティブであるときも同様である。

割り込み信号を受け付けると、CPUは特別なM1サイクルを発生する。

このM1サイクルの期間中に、IORQ信号が通常のMREQ信号に代ってアクティブとなる。これによって、割り込みの要求を出したデバイスからデータ・バス上に、８ビットのベクトルを乗せてもよい状態になる。これを割り込みアクノリッジという。

このサイクルには、自動的に待ち状態２サイクルが追加されるので注意を要する。

これらのサイクルの追加によって、リップル・プライオリティの割り込み方式が使用できるようになる。待ち状態として２サイクルあれば、リップル信号が安定に伝わり、入出力デバイスが出した応答ベクトルをCPUが確実にとり込むための時間としては、これで充分である。

第７章で、CPU内での割り込み応答ベクトルの処理形式について詳述してあるので参照されたい。

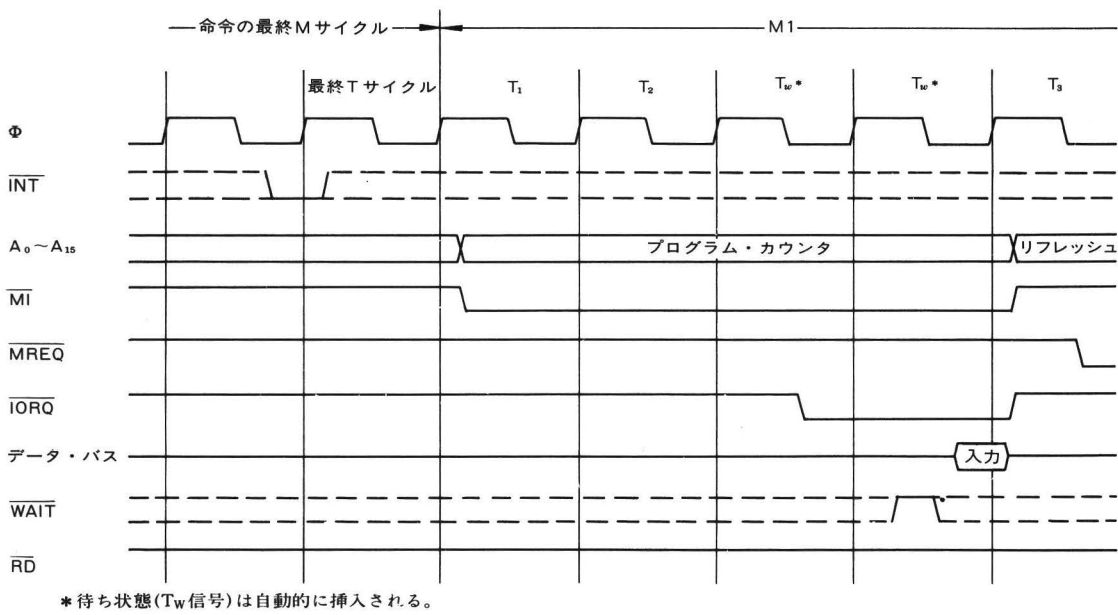



図3－6　割り込み要求／アクノリッジ・サイクル

図3－6A、図3－6Bにプログラマブル・カウンタを使用してアクノリッジ時間を延長する方法を示す。

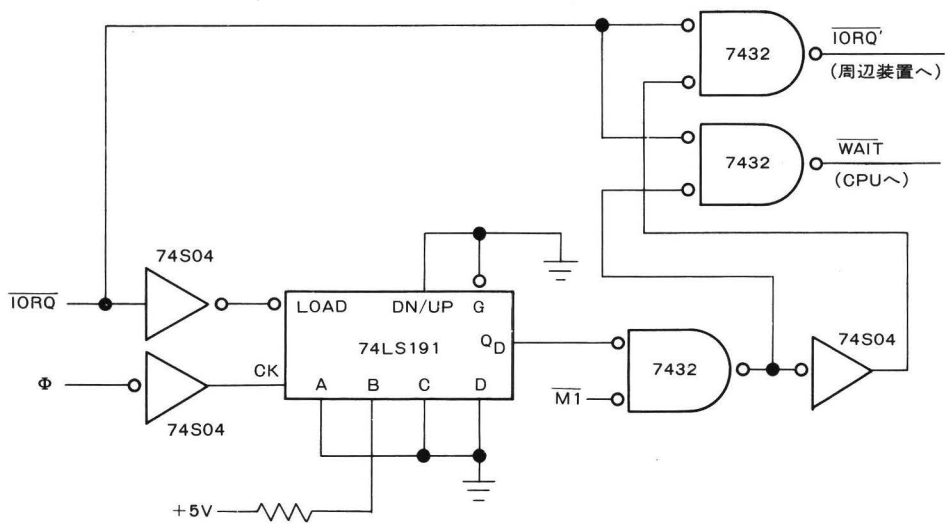



図3－6A　ウェイト・ステートを含む延長された割り込みアクノリッジ

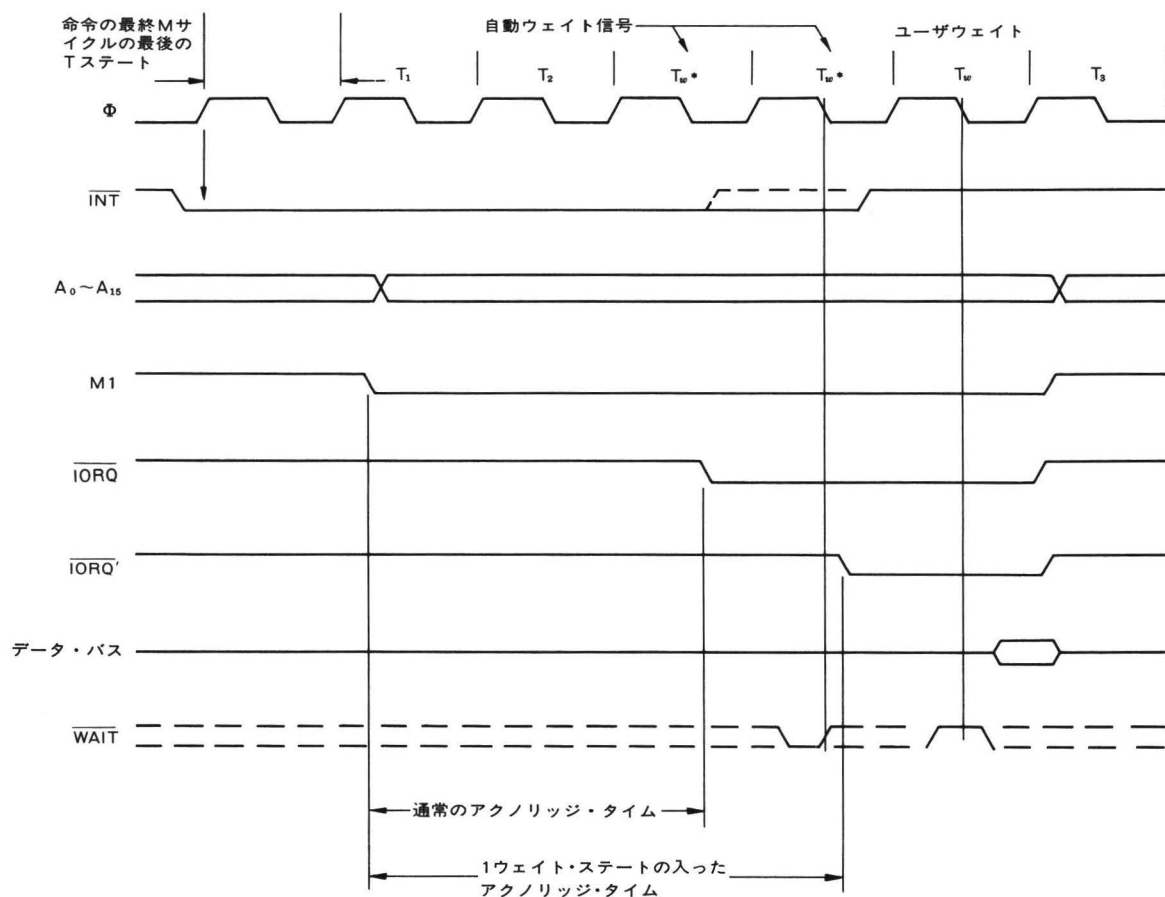



図3－6B　　ウェイト・ステートを1つ加えた要求／アクノリッジ・サイクル

3.6　ノン・マスカブル割り込みの応答

図3－7にノン・マスカブル割り込みのある場合の要求／アクノリッジ・サイクルの例を示す。

この信号はINTと異なり立ち下がりエッジが有効信号となっており、さらに各命令のどのタイミングで入力してもよい。ただし、現命令の実行完了後に割り込みアクノリッジ・サイクルに入るためには、少なくとも命令の最後のTサイクルまでに、NMIが立ち下がっていなければならない。この信号線はINT信号よりも優先順位が高く、またソフトウエアによってその機能を無効にすることもできない。

CPUはこのノン・マスカブル割り込みに対しては通常のメモリ読み出し動作と同じ応答をする。異なる点は、CPUがその時点でのデータ・バス上の内容を無視して自動的にPC（プログラム・カウンタ）の内容を外部スタックに退避した後、0066_H番地へジャンプする点である。

この割り込み方式を使用する場合、ノン・マスカブル割り込みのためのサービス・ルーチンは0066_H番地から書いておかねばならない。

3.7　ホールト状態解除

プログラム内のホールト命令ではCPUはNOP命令を実行し続けるが、割り込み入力があれば、この状態から抜け出る。(この場合の割り込みはノン・マスカブル割り込みか、割り込みフリップ・フロップがセットされている場合のマスカブル割り込みのいずれでもよい。)

これら2種の割り込み線を図3－8に示すように、各T_4サイクルのクロックの立ち上がりエッジで調べて上述のいずれかであれば、CPUは次のクロックの立ち上がりエッジで、ホールト状態から解除される。

次のサイクルで、受け付けた割り込みの種類に応じてそれぞれの割り込みのアクノリッジ・サイクルに入ることになる。

もし、同時にノン・マスカブルとマスカブルの割り込み信号が入った場合には、優先順位の高いノン・マスカブル割り込みが受け付けられる。

ホールト状態でNOP命令を実行させている理由は、メモリ用のリフレッシュ信号を出し続けるためである。

この状態での各サイクルは、メモリからのデータを無視する以外は通常のM1（フェッチ）サイクルと同様であり、CPU内部で自動的にNOP命令を実行している。ホールト命令を実行すると、CPUはホールト状態に入ったことを外部に対して知らせるためにHALT信号がアクティブとなる。

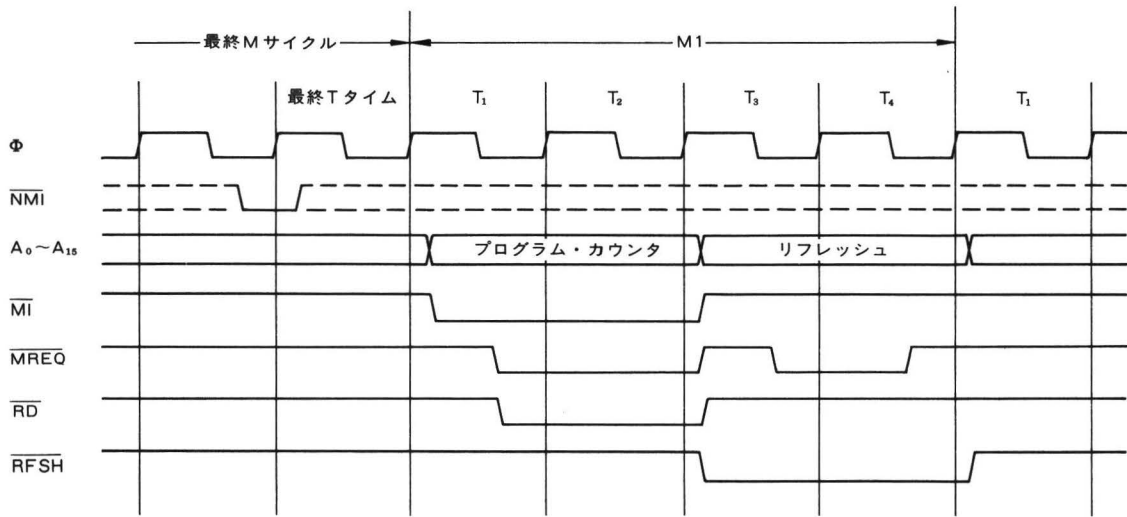



図3－7　　ノン・マスカブル割り込み、要求動作

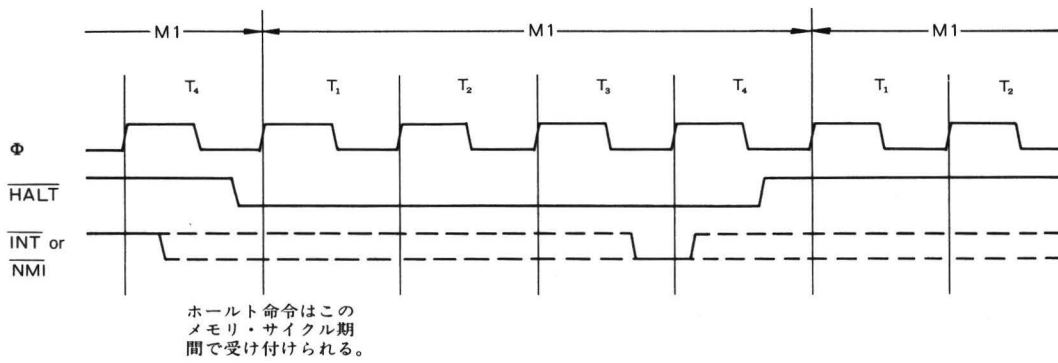



図3－8　　ホールト状態解除

4. 命令セット

Z－80A CPUには158種の命令があり、これらの命令には8080A CPUの78種の命令がすべて含まれている。
命令群は大別して次のようなグループに分けられる。

- ロード、交換
- ブロック転送、ブロック・サーチ
- 算術、論理処理
- ローテイト、シフト
- ビット操作（セット、リセット、テスト）
- ジャンプ、コール、リターン
- 入力、出力
- CPU制御

4.1 命令の型について

ロード命令は、データとCPU内部のレジスタ間で、あるいはCPUレジスタと外部メモリ間で、データを転送する機能を持っている。

この命令では、データを取り出すソースの番地と、それを格納するデスティネーションの番地を指定しなければならない。

ロード命令ではその実行後においてソースの内容は変化しない。

ロード命令の例として、レジスタCからレジスタBへデータを移すというような汎用レジスタ間のデータ転送命令がある。

また各CPUレジスタあるいは外部メモリに直接データをロードする命令や、CPUレジスタとメモリ間のデータの転送を行う命令がある。

交換命令は2つのレジスタ内のデータを入れ替える機能がある。

Z－80Aにはブロック転送というユニークな命令群がある。1命令で、どのようなサイズのメモリ・ブロックでも、メモリの他の領域に移し換えられる。このブロック転送の命令群は、大量のデータを処理する必要のあるときに非常に有効なものである。

Z－80Aのブロック・サーチ命令群も上記のような処理に有効である。1命令で必要な範囲だけ外部メモリのブロックをサーチし、特定の8ビット・キャラクタを探し出せる。目的のキャラクタが見つかれば命令は自動的に終了する。

これらの命令の実行中に割り込みをかけることは可能であり、長時間CPUが独占されてしまうという弊害はない。

算術および論理演算命令は、アキュムレータ内のデータとCPU汎用レジスタあるいは外部メモリ内のデータの操作を行うものである。

この操作の結果はアキュムレータに入り、その結果に応じて関係するフラグがセットされる。

算術演算命令の例として、アキュムレータと外部メモリの内容の加算命令がある。加算の結果はアキュムレータに入る。

これらの命令群のうちには、16ビットCPUレジスタ間の加減算命令などがある。

ローテイト、シフトの命令群によりどのようなレジスタまたはメモリであれ、キャリまたはキャリなしに、算術的または論理的に左右にローテイトできる。またアキュムレータ内の1ディジットとメモリ内の2ディジットで左右にローテイトすることもできる。

ビット操作命令は1命令でアキュムレータ、各汎用レジスタ、あるいは外部メモリ内のデータのどのビットでも、セット、リセットあるいはテストができるものである。たとえば、レジスタH内のデータの最上位ビットをリセットする、といったことが可能である。

この命令群は、制御への応用や一般のプログラミングにおいてフラグを多用する場合にとくに有効なものである。

ジャンプ、コール、リターン命令はユーザ・プログラムの流れを変えるためのものである。

この命令群にはいくつかの異なった形式があり、これらの形式により外部メモリの特定アドレスからプログラム・カウンタに更新データ（新しいアドレス値）を取ってくる形式も異なる。

ジャンプのうち、リスタート命令はユニークなもので、8ビットOPコードの一部に新アドレスを含んでいる。この命令では外部メモリのページ・ゼロ内の8箇所のアドレスを指定できる。

プログラム・ジャンプには、レジスタHL、IX、IYの内容を直接PCに移して行う方法もあり、ジャンプ先のアドレスをその時点で実行中のルーチンに関連させて決めることもできる。

Z－80Aの入出力命令群は、外部メモリあるいはCPU汎用レジスタ群と多数の外部入出力デバイス群とのデータ転送に適している。

入出力処理を行う際のポート番号はアドレス・バスの下位8ビットにより256ポートを指定できる。ある種の命令では、このポート番号を命令の2バイト目に持ち、また他の命令では、Cレジスタの内容で指定できるようになっている。

Cレジスタを入出力デバイスのポインタとして用いる方式の利点は、異なった入出力ポートでも共通の駆動ルーチンが使用できる点にある。これは、ルーチンがROMに入っていて、アドレスがOPコードの一部になっている場合には不可能である。

他の特長として、入力命令でフラグ・レジスタが自動的にセットされるので、入力データの状態（たとえば、入力のパリティ）を決めるのに余分の命令を使用する必要がない点があげられる。

Z－80A CPUには、自動的に256バイト以上のブロック・データをメモリのある領域から入出力ポートへ、またその逆に転送する命令もある。

対になっている汎用レジスタ群を両方使用して、これらの命令により高速の入出力ブロック転送ができる。

この入出力命令群の真価は次の例でよくわかるだろう。割り込み駆動形式を用いて倍密度フロッピー・ディスクを動作させる場合、Z－80A CPUは必要なフロッピー・ディスク・フォーマット（すなわち、プレ・アンブル、アドレス、データ、CRCコード使用・未使用など）をプログラムによってたやすく用意できる。

最後のCPU制御命令は種々の選択、モード設定のためのものである。この命令群には割り込みイネーブル・フリップ・フロップをセット、リセットしたり、割り込み応答のモードを設定したりする命令などがある。

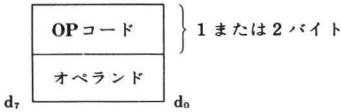
4.2 アドレシング・モード

Z－80A命令群には、CPU内部レジスタ、外部メモリあるいは入出力ポート内にストアされたデータの操作命令が多く設けられている。

これらのデータのある番地の指定を行うことをアドレシングという。

この節では、Z－80Aで使用されるアドレシングのタイプについて簡単な説明をし、あとで各命令群の説明を行う際にさらに詳しく述べる。

イミディエット・アドレシング　このアドレシング・モードでは、OPコードに続く1バイトを実効オペランドとする。

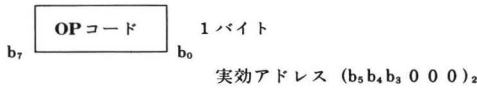
d_7 〜 d_0	
OPコード	1または2バイト
オペランド	

例として、アキュムレータに定数をロードするような命令がある。

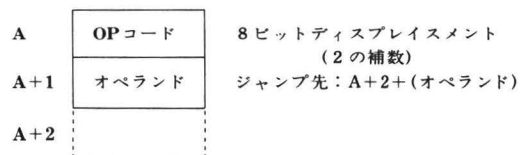
OPコードに続く1バイトが直接その定数値となる。

拡張イミディエット・アドレシング　このモードは上述のモードを拡張したものであり、オペランドが2バイトになっているものである。

OPコード	1または2バイト
オペランド	下位データ
オペランド	上位データ

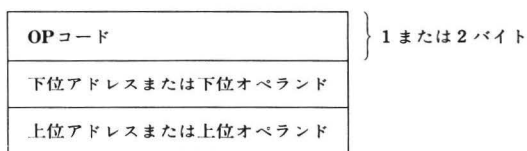
たとえば、16ビット（2バイト）データをHLレジスタ対にロードする命令がある。

ゼロ・ページ修飾アドレシング　Z－80Aには、メモリのゼロ・ページの8箇所を1バイトで指定する特別なコール命令がある。これはリスタート命令と呼ばれ、PCをゼロ・ページの実効アドレスにセットするものである。この命令の値は、共通に使用するサブルーチンのアドレス（16ビット）を1バイトで指定できるようになっていて、ステップ数を節約できる。

相対アドレシング　このモードでは、OPコードに続く1バイトにより、ジャンプ命令のある場所からのディスプレイスメントを指定する。このディスプレイスメントは符号付の2の補数であり、次の命令の先頭番地の値にこのディスプレイスメントを加えた値がジャンプ先の番地となる。

アドレス	内容	
A	OPコード	8ビットディスプレイスメント（2の補数）
A+1	オペランド	ジャンプ先：A+2+(オペランド)
A+2		

このモードは近隣のアドレスへ2バイトでジャンプでき、3バイトのジャンプ命令を使用する必要がないときに役に立つ。多くの場合、関係のあるプログラムのセグメントは近接して置かれるのが普通でありこのモードを利用してメモリの節約が図れる。

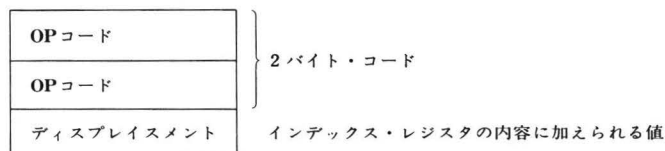
符号付2の補数の範囲は−128から+127であり、これをA+2に対して加えるのでOPコードのアドレスから計算すると、+129から−126の範囲へのジャンプ指定となる。

このモードの最大の特長はリロケータブル・コードとなる点である。

拡張アドレシング　このアドレシング・モードは命令に2バイトのアドレス・データを含む。64Kバイト内のどのアドレスに対しても、プログラムをジャンプさせられる。

拡張アドレシングはメモリのある位置から他の位置へジャンプしたり、ある位置でのデータの出し入れの命令に必要なものである。

このモードにおいて、ソースあるいはデスティネーション・アドレスを指定するオペランドとしてこれを（nn）と書いたとき、nn番地のメモリの内容を意味する。ここで、nnは16ビットのアドレス指定値であり、括弧の中のnnは2バイトのメモリ番地ポインタである。

たとえば（1200）は1200番地のメモリの内容である。

インデックス・アドレシング　このアドレシングでは、OPコードのあとにはメモリ・アドレスのポインタである2つのインデックス・レジスタのうち、いずれか一方（OPコードで指定する）の内容に加えるディスプレイスメントを指定するデータ・バイトが置かれる。この操作ではインデックス・レジスタの内容は変らない。

インデックス・アドレシング命令の例として、アキュムレータにメモリ（インデックス+ディスプレイスメント）の内容をロードする命令がある。ディスプレイスメントは符号付の2の補数である。

インデックス・モードの特長として、データ・テーブルを使用する場合、インデックス・レジスタにはどのテーブルのスタート・アドレスでも入れておけるので、非常に簡単に使用できるという点がある。2つのインデックス・レジスタがあるので、2つ以上のテーブルをよく使用する必要があるときには都合がよい。インデックス・モードも相対アドレシングと同様にリロケータブル・コードである。

インデックス・レジスタとしてIX、IYがある。インデックス・アドレシングでは次のような書き方が使用される。

（IX+d）または（IY+d）

ここで、dはOPコードのあとに置かれるディスプレイスメントである。括孤はこの中の値が外部メモリのポインタとして使用されていることを意味している。

レジスタ・アドレシング　Z−80Aの多くのOPコードにはレジスタ指定ビットがあり、どのCPUレジスタを操作用として使用するかを指定するようになっている。レジスタ・アドレシングの例として、Bレジスタの内容をCレジスタにロードする命令などがある。

インプライド・アドレシング　このモードでは、OPコード自体がレジスタ指定を含んでいて、1つまたは2つのレジスタが自動的

に指定される。

たとえば、算術演算命令では、つねにアキュムレータが結果をセットするレジスタとなる。

レジスタ間接アドレシング　このタイプでは、メモリ内の位置指定のポインタとして16ビットのレジスタ・ペア（例：HL）が使用される。これは大変強力で役に立つ命令である。

OPコード	1または2バイト

例として、HLレジスタの内容をメモリの位置ポインタとし、その位置の内容をアキュムレータにロードするといった命令がある。

インデックス・アドレシングは、インデックス・レジスタの内容にディスプレイスメントを加えるといった操作を含んでいる以外は、レジスタ間接アドレシング・モードの一形式である。

ブロック転送、ブロック・サーチの命令はこのモードの拡張形であり、レジスタの内容の自動インクリメント、デクリメントおよび比較機能が付加されている。間接モードでの記法はレジスタ名を括弧で囲む。そしてそのレジスタをポインタとする。メモリ位置ポインタがHLレジスタの内容である場合、

(HL)

と書く。間接モードでは16ビット・オペランドをよく使用するがこの場合、レジスタの内容はまず、オペランドの下位バイトを指し、次にレジスタの内容が自動的にインクリメントされ、上位バイトを指す。

ビット・アドレシング　Z－80Aには、多くのビット・セット、リセットおよびテスト命令がある。これらの命令は、どのメモリ位置、CPUレジスタに対しても有効で、前述した3モード（レジスタ、レジスタ間接、およびインデックス）のいずれかのアドレシング・モードでビット操作ができる。

8ビットのうちのどのビットを操作するかはOPコード内の3ビットにより指定する。

アドレシング・モードの組み合わせ

これらの命令には、算術演算やロード命令のように、1つ以上のオペランドが含まれている命令が多い。これらの命令では、2つのアドレシング・モードを組み合わせて用いる。たとえば、ロード命令で、ソースの指定はイミディエット、デスティネーションの指定はレジスタ間接あるいはインデックス・アドレシングで行う場合などである。

4.3　命令OPコード

この節では、Z－80Aの各命令について説明し、それらのOPコード表を示す。これら各表の陰影部で示したOPコードは8080Aのものと同一である。

各命令に対応するアセンブリ語ニーモニックを同時に示す。OPコードは16進数表現にしてあり、1バイトのコードは2桁の16進数、2バイトでは4桁の16進数で書く。16進から2進への変換は次の表を利用するとよい。

16進数		2進数		10進数	16進数		2進数		10進数
0	=	0000	=	0	8	=	1000	=	8
1	=	0001	=	1	9	=	1001	=	9
2	=	0010	=	2	A	=	1010	=	10
3	=	0011	=	3	B	=	1011	=	11
4	=	0100	=	4	C	=	1100	=	12
5	=	0101	=	5	D	=	1101	=	13
6	=	0110	=	6	E	=	1110	=	14
7	=	0111	=	7	F	=	1111	=	15

Z－80Aの命令ニーモニックとしてOPコード単独のものと、OPコードと1つまたは2つのオペランドより構成されているものがある。オペランドのない命令はオペランドがOPコードに内蔵されていると考える。

1つの論理オペランドのみの場合、あるいはロジカルORのような片方のオペランドが不定の場合には、1オペランド・ニーモニックになっている。

2変数を扱う命令は2オペランド・ニーモニックである。

ロードと交換命令（LD, PUSH, POP）

表4－1にCPUの8ビット・ロード命令群を示す。各命令で使用されるアドレシングの型も併記しておく。データのソースは上上欄に、デスティネーションは左側端に示してある。例として、BレジスタからCレジスタへのロードはOPコード48_Hとなる。

表中のコードはすべて16進数表現であるが、この48_H(2進数0100 1000)コードはM1サイクル時に外部メモリからCPUにフェッチされ、CPUはこれをデコードしてレジスタ間のデータの転送を実行する。

この命令群のアセンブリ語ニーモニックはLDであり、これのあとにデスティネーション、ソースの順で書く。(LD DEST., SOURCE)

これらの命令ではいくつかのアドレシング・モードを組み合わせて使用できる。たとえば、ソースをレジスタ・モード、デスティネーションをレジスタ間接モードとした命令では次のように記述できる。

LD (HL), D

これはHLレジスタで指定されたメモリへDレジスタの内容をロード(移す)するニーモニック命令で、OPコードは72_Hである。

ここでHLを囲んだ括弧は、HLの内容をメモリの位置ポインタとして使う、ということを意味している。

Z−80Aのアセンブリ言語はプログラミングが容易なようにできている。

各命令は自己説明的(セルフ・ドキュメンティング)で意味がわかりやすく、プログラムの保守も楽である。

Z−80Aで使用されるOPコードには、2バイトのものがいくつかあるので注意されたい。

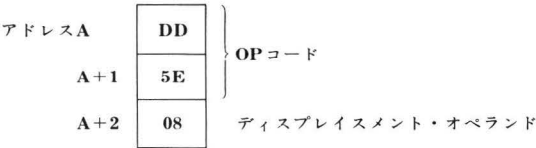
命令には8ビット、16ビット、24ビット、32ビットの命令があるが、これはメモリの有効利用に役立つ。つまり、算術、論理演算のような命令で、8ビットで済む場合にでも、16ビット固定の命令を使用しなければならないとしたら、メモリのむだになるからである。

LD命令で、ソースあるいはデスティネーションの位置のアドレシングにインデックス・モードを使用した場合、メモリは3バイト必要で、3バイト目はディスプレイスメントdとなる。

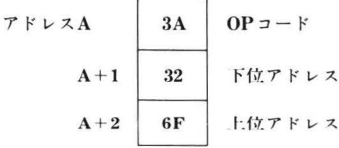
たとえば、次のような命令ニーモニックは、IXの内容から8バイト目の場所の内容をレジスタEにロードするもので、数字8がオフセットでありディスプレイスメントとなる。

LD E, (IX+8)

命令のメモリ上での順序は次のようになる。

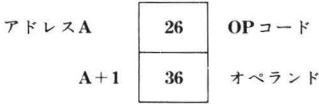
アドレスA	DD	OPコード
A+1	5E	
A+2	08	ディスプレイスメント・オペランド

拡張アドレシング命令も3バイトである。メモリ$6F32_H$番地の内容をアキュムレータにロードする次のような命令では、メモリ上の順序は次のようになる。

LD A, (6F32H)

アドレスA	3A	OPコード
A+1	32	下位アドレス
A+2	6F	上位アドレス

アドレスの下位バイトはつねに第1オペランドである点に注意しなければならない。

汎用8ビット・レジスタへ直接ロードする命令は2バイト命令である。値36_HをレジスタHへロードする命令は次のように書ける。

LD H, 36H

アドレスA	26	OPコード
A+1	36	オペランド

デスティネーション側のメモリをインデックス・アドレシングで指定し、ソース側をイミディエットで指定してロードする場合、4バイトを必要とする。

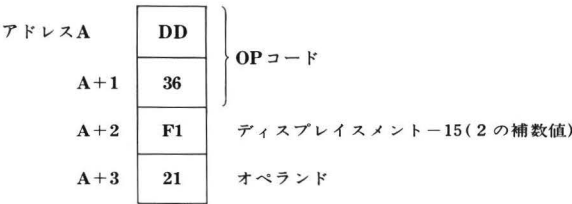
LD (IX−15), 21H

インデックス・アドレシングでは、いつもディスプレイスメントはOPコードの直後にあるので注意を要する。

表4−2に16ビット・ロード命令群のOPコードを示す。これは8ビットのものとよく似ている。注意すべき点は、拡張アドレシングは全部のレジスタ・ペアに対して使用できることである。

さらに、スタック・ポインタを用いたレジスタ間接アドレシング命令とし 、PUSH と POP 命令がある。これらのニーモニックはそれぞれ〝PUSH〞、〝POP〞である。この命令ではスタックへあるいはスタックからプッシュやポップする際、スタックポインタが自動的にインクリメントあるいはデクリメントされる点が、他の16ビット命令と異なっている。

PUSH AF

これはOPコード$F5_H$の1バイト命令であるが、次のような操作が自動的に行われる。

○(SP) を−1する。
○LD (SP), A
○(SP) を−1する。
○LD (SP), F

この結果、外部スタックは次のようになる。

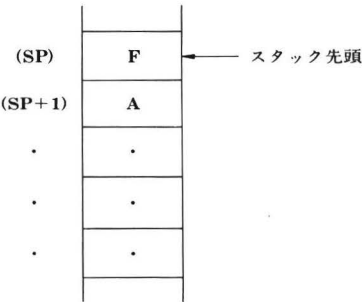
ソース

デスティネーション		インプライド		レジスタ										インデックス			イミディエット
		I	R	A	B	C	D	E	H	L	(HL)	(BC)	(DE)	(IX+d)	(IY+d)	(nn)	n
レジスタ	A	ED 57	ED 5F	7F	78	79	7A	7B	7C	7D	7E	0A	1A	DD 7E d	FD 7E d	3A n n	3E n
	B			47	40	41	42	43	44	45	46			DD 46 d	FD 46 d		06 n
	C			4F	48	49	4A	4B	4C	4D	4E			DD 4E d	FD 4E d		0E n
	D			57	50	51	52	53	54	55	56			DD 56 d	FD 56 d		16 n
	E			5F	58	59	5A	5B	5C	5D	5E			DD 5E d	FD 5E d		1E n
	H			67	60	61	62	63	64	65	66			DD 66 d	FD 66 d		26 n
	L			6F	68	69	6A	6B	6C	6D	6E			DD 6E d	FD 6E d		2E n
レジスタ間接	(HL)			77	70	71	72	73	74	75							36 n
	(BC)			02													
	(DE)			12													
インデックス	(IX+d)			DD 77 d	DD 70 d	DD 71 d	DD 72 d	DD 73 d	DD 74 d	DD 75 d							DD 36 d n
	(IY+d)			FD 77 d	FD 70 d	FD 71 d	FD 72 d	FD 73 d	FD 74 d	FD 75 d							FD 36 d n
拡張アドレシング	(nn)			32 n n													
インプライド	I			ED 47													
	R			ED 4F													

表4－1　8ビットロード　〝LD〞

ポップ命令はプッシュ命令とちょうど逆の操作になる。

これらの命令では、オペランドは16ビットであり、上位バイトが最初にプッシュされ、また最後にポップされる。

PUSH　BC　は　PUSH　B　そして　C
PUSH　DE　は　PUSH　D　そして　E
PUSH　HL　は　PUSH　H　そして　L
POP　HL　は　POP　L　そして　H

となる。

ソースに対する拡張イミディエット・アドレシングでは、OPコードのあとに2バイトのデータが付く。

LD DE, 0659H

アドレスA	11	OPコード
A+1	59	下位バイト(レジスタEへ入る)
A+2	06	上位バイト(レジスタDへ入る)

すべての拡張イミディエットあるいは拡張モードでは、下位バイトがOPコードのすぐあとに続く。

表4－3にZ－80Aに設けられている16ビット・データの交換命令を示す。

OPコード08_Hは、2組のアキュムレータとフラグ類の切り換えをプログラムで実行させるためのもので、また$D9_H$で、6個の汎用レジスタ群をもう一方のレジスタ群に切り換えることができる。

これらのOPコードは、これ以下に短くできないという最短の1バイト命令であり、2セットのレジスタ群を即時切り換えて使用できるので、高速の割り込み応答が必要な場合に利用できる。

ブロック転送およびサーチ

表4－4に、非常に特長があり、しかも強力なブロック転送命令を示す。これらの操作は次の3つのレジスタを使用して行う。

HL　ソースの位置のポインタ
DE　デスティネーションのポインタ
BC　バイト・カウンタ

プログラムでは、これらのいずれかのブロック転送命令を使用するまえに、上記の3つのレジスタに値をセットしておく必要がある。

LDI（ロード&インクリメント）命令は、(HL) で指定される位置から (DE) で指定される位置へ、1バイトだけ転送する。その後、2つのレジスタの内容はインクリメントされて次の位置を指している。

バイト・カウンタ（BC）はデクリメントされる。この命令は、データ転送の途中で他の処理を行いたい場合に便利である。

LDIR（ロード&インクリメント&リピート）命令はLDIの拡張形で、自動的に全部（バイト・カウンタが零になるまで）データを転送してしまいたい場合に便利である。

16ビット・レジスタを使用するので、ブロックの大きさは最大64Kバイト（1Kバイト＝1024バイト）まで指定でき、ソース、デスティネーションのアドレスはメモリ空間のどこにでも配置できる。

さらに、ブロック相互のオーバラップも可能で、3つのレジスタ・ペアを使用して扱えるデータに何ら制限のないことに注目されたい。

LDDとLDDRはさきの2つとよく似ている。ただ、データ転送後、レジスタ・ペアHL、DEがデクリメントされる点が異なっていて、位置の高い方から低い方へ転送する場合に使用するのに適している。

		ソース									
		レジスタ							拡張イミディエット	拡張アドレシング	レジスタ間接
		AF	BC	DE	HL	SP	IX	IY	nn	(nn)	(SP)
デスティネーション レジスタ	AF										F1
	BC								01 n n	ED 4B n n	C1
	DE								11 n n	ED 5B n n	D1
	HL								21 n n	2A n n	E1
	SP				F9		DD F9	FD F9	31 n n	ED 7B n n	
	IX								DD 21 n n	DD 2A n n	DD E1
	IY								FD 21 n n	FD 2A n n	FD E1
拡張アドレシング	(nn)		ED 43 n n	ED 53 n n	22 n n	ED 73 n n	DD 22 n n	FD 22 n n			
PUSH命令 → レジスタ間接	(SP)	F5	C5	D5	E5		DD E5	FD E5			

POP命令 ↑（(SP)列）

(注) PUSHおよびPOP命令の全実行が終わったあとにSPが修正される。

表4－2　16ビットロード〝LD″

		インプライド・アドレシング				
		AF′	BC′, DE′ & HL′	HL	IX	IY
インプライド	AF	08				
	BC, DE & HL		D9			
	DE			EB		
レジスタ間接	(SP)			E3	DD E3	FD E3

表4－3　交換

HLはソースのポインタ
DEはデスティネーションのポインタ
BCはバイトカウンタ

		ソース レジスタ間接 (HL)	
デスティネーション レジスタ間接	(DE)	ED A0	"LDI"－Load (DE)←(HL) Inc HL & DE, Dec BC
		ED B0	"LDIR"－Load (DE)←(HL) Inc HL & DE, Dec BC, Repeat until BC=0
		ED A8	"LDD"－Load (DE)←(HL) Dec HL & DE, Dec BC
		ED B8	"LDDR"－Load (DE)←(HL) Dec HL & DE, Dec BC, Repeat until BC=0

表4－4　ブロック転送グループ

HLはアキュムレータの内容と照合するメモリの位置のポインタ
BCはバイトカウンタ

サーチロケーション レジスタ間接 (HL)	
ED A1	"CPI" Inc HL, Dec BC
ED B1	"CPIR" Inc HL, Dec BC repeat until BC=0 or find match
ED A9	"CPD" Dec HL & BC
ED B9	"CPDR" Dec HL & BC Repeat until BC=0 or find match

表4－5　ブロックサーチ

表４－５に４種のブロック・サーチ命令のOPコードを示す。

CPI（コンペア&インクリメント）は、HLレジスタで指定されるメモリの内容とアキュムレータ内のデータを比較し、その結果をフラグ・ビット（第5章を参照）の1つにセットする。その後、HLレジスタ・ペアをインクリメントし、バイト・カウンタ（BCレジスタ・ペア）をデクリメントする。

CPIRはCPI命令の拡張形であり、バイト・カウンタが零になるまで比較を繰り返す。この1命令で、ある8ビット・キャラクタを全メモリ内で探すこともできる。

CPD（コンペア&デクリメント）とCPDR（コンペア&デクリメント&リピート）は同様な命令であるが、各比較ののちHLレジスタをデクリメントする点のみが異なっている。したがって、逆方向へのメモリのサーチが行える。(このサーチはメモリ・ブロックの最高アドレス近くからスタートし、下位アドレスへすすむ。)

このブロック転送、サーチ命令は、文字列の操作などの応用に非常に強力な道具となる。

算術と論理演算

表4－6にアキュムレータを中心にした8ビット算術演算命令群を示す。インクリメント（INC)、デクリメント（DEC）命令もともに示す。

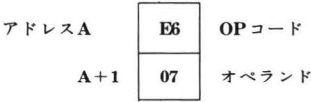
これらのうち、INC、DECを除いた8ビット操作命令で、アキュムレータと表に示したソース・データとの間の操作が行われ、その結果は、コンペア（CP）命令を除き、アキュムレータに置かれる。CPでは命令実行の前後でアキュムレータの内容は変わらない。これらの操作の結果で、フラグ・レジスタがセットされる。(各命令でセットされるフラグの詳細については、第5章に述べる。)

INC、DEC命令は、レジスタはどのメモリの位置に対しても実行できるが、ソースとデスティネーションを別々にとることはできない。

ソース・オペランド（ソースの指定）にインデックス・レジスタを使用した場合、ディスプレイスメントをすぐあとに続けて指定しておく。

イミディエット・アドレシングを使用した場合には、実際のオペランドで直接指定する。例として、次のような命令では、あとのようなメモリ配置となる。

AND　07H

アドレスA	E6	OPコード
A+1	07	オペランド

アキュムレータの内容が$F3_H$であれば、結果として03_Hが得られ、最終的にアキュムレータの内容は03_Hとなる。

ソース

	レジスタ・アドレシング							レジスタ間接	インデックス		イミディエット
	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)	n
"ADD"	87	80	81	82	83	84	85	86	DD 86 d	FD 86 d	C6 n
ADD w CARRY "ADC"	8F	88	89	8A	8B	8C	8D	8E	DD 8E d	FD 8E d	CE n
SUBTRACT "SUB"	97	90	91	92	93	94	95	96	DD 96 d	FD 96 d	D6 n
SUB w CARRY "SBC"	9F	98	99	9A	9B	9C	9D	9E	DD 9E d	FD 9E d	DE n
"AND"	A7	A0	A1	A2	A3	A4	A5	A6	DD A6 d	FD A6 d	E6 n
"XOR"	AF	A8	A9	AA	AB	AC	AD	AE	DD AE d	FD AE d	EE n
"OR"	B7	B0	B1	B2	B3	B4	B5	B6	DD B6 d	FD B6 d	F6 n
COMPARE "CP"	BF	B8	B9	BA	BB	BC	BD	BE	DD BE d	FD BE d	FE n
INCREMENT "INC"	3C	04	0C	14	1C	24	2C	34	DD 34 d	FD 34 d	
DECREMENT "DEC"	3D	05	0D	15	1D	25	2D	35	DD 35 d	FD 35 d	

表4－6　8ビット算術、論理演算

操作	コード
Decimal Adjust Acc, "DAA"	27
Complement Acc, "CPL"	2F
Negate Acc, "NEG" (2's complement)	ED 44
Complement Carry Flag, "CCF"	3F
Set Carry Flag, "SCF"	37

表4－7　その他の操作

AND　0 H の操作と結果

操作前のACCの内容	1111	0011	＝	$F3_H$
オペランド	0000	0111	＝	07_H
操作後のACCの内容	0000	0011	＝	03_H

ADD命令(ADD)は、ソースの位置内のデータとアキュムレータ(ACC)のデータ間で2進加算を行い、SUB命令は2進減算を行う命令である。

キャリを含めた操作を行う場合、ADC（加算)、SBC（減算）を使用すればよい。それぞれキャリ・フラグの内容を演算に関与させられる。

Z－80Aでは、フラグとデシマル・アジャスト（10進補正）の命令を次のような算術演算操作に対して使用できる。

パックされたBCD数の高精度演算
符号付2進数の高精度演算
符号付2の補数値の高精度演算

この命令群には他に論理積AND、論理和OR、排他的論理和XOR、比較CPがある。

アキュムレータあるいは、キャリ・フラグを操作する5つの汎用算術演算命令があるが、これを表5－7に示す。
10進補正は加算と同様減算に対しても行うことができ、BCD算術演算が簡単に行える。この減算補正にはNフラグを利用するとよい。このフラグは前の算術演算命令が減算であればセットされている。
NEG命令はアキュムレータ内の値の2の補数値をアキュムレータに置く（セットする）。
ここで、Z－80Aにはキャリをリセットする命令がないことに注意されたい。この操作はアキュムレータ自身でANDをとるなどの操作をすれば、キャリ・フラグがリセットされるので、その命令をとくに設けていない。

表4－8に16ビット・レジスタ間の16ビット演算命令群を示す。これらにはキャリを含めた加、減算を入れて、5グループある。
ADC、SBC命令は全フラグに影響を与える。これらの命令を使用すればアドレス計算あるいは16ビット算術演算操作の簡単化が図れる。

デスティネーション ＼ ソース		BC	DE	HL	SP	IX	IY
"ADD"	HL	09	19	29	39		
	IX	DD 09	DD 19		DD 39	DD 29	
	IY	FD 09	FD 19		FD 39		FD 29
ADD WITH CARRY AND SET FLAGS "ADC"	HL	ED 4A	ED 5A	ED 6A	ED 7A		
SUB WITH CARRY AND SET FLAGS "SBC"	HL	ED 42	ED 52	ED 62	ED 72		
INCREMENT "INC"		03	13	23	33	DD 23	FD 23
DECREMENT "DEC"		0B	1B	2B	3B	DD 2B	FD 2B

表4－8　　16ビット算術演算

ローテイトとシフト

Z－80Aの主な特長として、アキュムレータ、汎用レジスタあるいはメモリ内のデータのローテイトあるいはシフト命令が充実している点があげられる。
ローテイトとシフト命令のOPコードを表4－9に示す。Z－80Aには、算術および論理シフト操作も設けられている。これらの操作は整数の乗算および除算を含む広範囲の用途に使用でき、大層便利なものである。
2つのBCDディジット・ローテイト命令（RRDとRLD）は、HLレジスタ・ペアで指定されるメモリ内の2桁（ディジット、4ビットで1桁を表わす）とアキュムレータ内の1桁とのローテイト（入れ替え）が行える。（表4－9を参照）これらの命令はBCDデータの演算に利用できる。

ソースおよびデスティネーション

ローテイトあるいはシフトの型	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
"RLC"	CB 07	CB 00	CB 01	CB 02	CB 03	CB 04	CB 05	CB 06	DD CB d 06	FD CB d 06
"RRC"	CB 0F	CB 08	CB 09	CB 0A	CB 0B	CB 0C	CB 0D	CB 0E	DD CB d 0E	FD CB d 0E
"RL"	CB 17	CB 10	CB 11	CB 12	CB 13	CB 14	CB 15	CB 16	DD CB d 16	FD CB d 16
"RR"	CB 1F	CB 18	CB 19	CB 1A	CB 1B	CB 1C	CB 1D	CB 1E	DD CB d 1E	FD CB d 1E
"SLA"	CB 27	CB 20	CB 21	CB 22	CB 23	CB 24	CB 25	CB 26	DD CB d 26	FD CB d 26
"SRA"	CB 2F	CB 28	CB 29	CB 2A	CB 2B	CB 2C	CB 2D	CB 2E	DD CB d 2E	FD CB d 2E
"SRL"	CB 3F	CB 38	CB 39	CB 3A	CB 3B	CB 3C	CB 3D	CB 3E	DD CB d 3E	FD CB d 3E
"RLD"								ED 6F		
"RRD"								ED 67		

	A
RLCA	07
RRCA	0F
RLA	17
RRA	1F

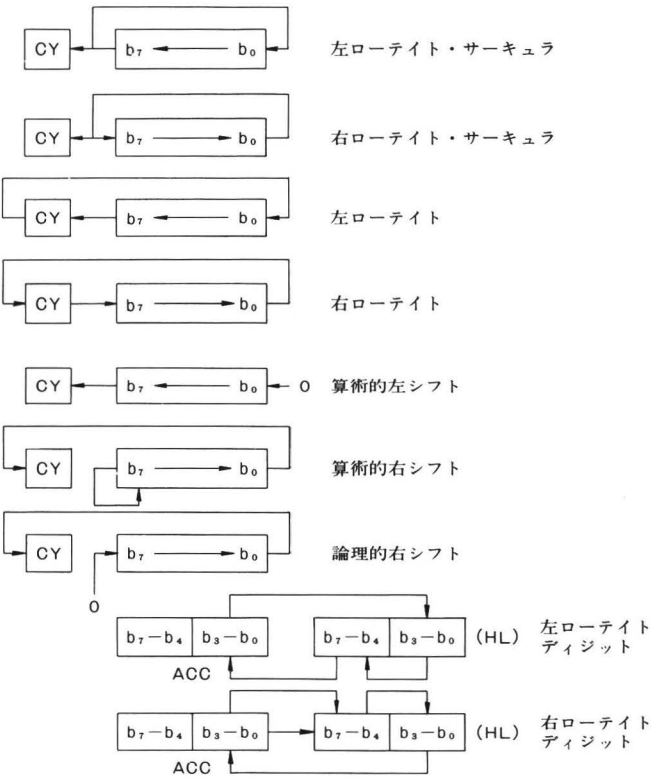



表4－9　ローテイトとシフト

ビット操作

ビットごとのセット、リセットあるいはテストといった操作はどのようなプログラムでも必要なものである。とくにその操作がレジスタに限らずメモリ内でできれば、大変便利である。このビットは普通汎用ソフトウエア・ルーチン、外部の制御状態の指示

あるいはメモリの有効利用のためにパックしておくデータ群の指標などのためのフラグとしてよく使用される。

Z－80Aには、1命令によりアキュムレータ、各汎用レジスタ、あるいはどのメモリ位置に対してもその中の各ビットのセット、リセット、あるいはテストができる能力がある。

表4－10にこの目的のための240の命令群を示す。

レジスタ・アドレシングでは、アキュムレータあるいはいずれかの汎用レジスタを指定でき、そこでのビット操作ができる。

レジスタ間接およびインデックス・アドレシングでは、その指定によって外部メモリ上でビット操作ができる。

ビット・テスト操作では、ビットの状態をゼロ・フラグ（Z）で判定する。たとえば、Zフラグがセットされているならば、テストされるビットは零である。

	ビット	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
		レジスタ・アドレシング							レジスタ間接	インデックス	
TEST "BIT"	0	CB 47	CB 40	CB 41	CB 42	CB 43	CB 44	CB 45	CB 46	DD CB d 46	FD CB d 46
	1	CB 4F	CB 48	CB 49	CB 4A	CB 4B	CB 4C	CB 4D	CB 4E	DD CB d 4E	FD CB d 4E
	2	CB 57	CB 50	CB 51	CB 52	CB 53	CB 54	CB 55	CB 56	DD CB d 56	FD CB d 56
	3	CB 5F	CB 58	CB 59	CB 5A	CB 5B	CB 5C	CB 5D	CB 5E	DD CB d 5E	FD CB d 5E
	4	CB 67	CB 60	CB 61	CB 62	CB 63	CB 64	CB 65	CB 66	DD CB d 66	FD CB d 66
	5	CB 6F	CB 68	CB 69	CB 6A	CB 6B	CB 6C	CB 6D	CB 6E	DD CB d 6E	FD CB d 6E
	6	CB 77	CB 70	CB 71	CB 72	CB 73	CB 74	CB 75	CB 76	DD CB d 76	FD CB d 76
	7	CB 7F	CB 78	CB 79	CB 7A	CB 7B	CB 7C	CB 7D	CB 7E	DD CB d 7E	FD CB d 7E
RESET BIT "RES"	0	CB 87	CB 80	CB 81	CB 82	CB 83	CB 84	CB 85	CB 86	DD CB d 86	FD CB d 86
	1	CB 8F	CB 88	CB 89	CB 8A	CB 8B	CB 8C	CB 8D	CB 8E	DD CB d 8E	FD CB d 8E
	2	CB 97	CB 90	CB 91	CB 92	CB 93	CB 94	CB 95	CB 96	DD CB d 96	FD CB d 96
	3	CB 9F	CB 98	CB 99	CB 9A	CB 9B	CB 9C	CB 9D	CB 9E	DD CB d 9E	FD CB d 9E
	4	CB A7	CB A0	CB A1	CB A2	CB A3	CB A4	CB A5	CB A6	DD CB d A6	FD CB d A6
	5	CB AF	CB A8	CB A9	CB AA	CB AB	CB AC	CB AD	CB AE	DD CB d AE	FD CB d AE
	6	CB B7	CB B0	CB B1	CB B2	CB B3	CB B4	CB B5	CB B6	DD CB d B6	FD CB d B6
	7	CB BF	CB B8	CB B9	CB BA	CB BB	CB BC	CB BD	CB BE	DD CB d BE	FD CB d BE
SET BIT "SET"	0	CB C7	CB C0	CB C1	CB C2	CB C3	CB C4	CB C5	CB C6	DD CB d C6	FD CB d C6
	1	CB CF	CB C8	CB C9	CB CA	CB CB	CB CC	CB CD	CB CE	DD CB d CE	FD CB d CE
	2	CB D7	CB D0	CB D1	CB D2	CB D3	CB D4	CB D5	CB D6	DD CB d D6	FD CB d D6
	3	CB DF	CB D8	CB D9	CB DA	CB DB	CB DC	CB DD	CB DE	DD CB d DE	FD CB d DE
	4	CB E7	CB E0	CB E1	CB E2	CB E3	CB E4	CB E5	CB E6	DD CB d E6	FD CB d E6
	5	CB EF	CB E8	CB E9	CB EA	CB EB	CB EC	CB ED	CB EE	DD CB d EE	FD CB d EE
	6	CB F7	CB F0	CB F1	CB F2	CB F3	CB F4	CB F5	CB F6	DD CB d F6	FD CB d F6
	7	CB FF	CB F8	CB F9	CB FA	CB FB	CB FC	CB FD	CB FE	DD CB d FE	FD CB d FE

表4－10　　ビット操作グループ

ジャンプ、コール、リターン

表4－11にジャンプ、コール、およびリターン命令群を示す。ジャンプとはプログラム中でのステップの変更であり、プログラム・カウンタには、3つの適当なアドレシング・モード（拡張イミディエット、相対、またはレジスタ間接）のいずれかで指定される16ビットの値がロードされる。

ジャンプ命令群には条件ジャンプがある。これらはジャンプ命令直前の条件によってジャンプしたり、そのまま次の命令ステップへ続けられるものである。この条件はフラグ・レジスタ内のデータにセットされている。(フラグ・レジスタに関しては、第5章を参照)

拡張イミディエット・アドレシングを使用すれば、メモリのどの位置へでもジャンプできる。この命令は3バイト必要で、OPコードの次のバイトが下位アドレス・バイトであり、そのあとに上位アドレス・バイトが続く。(2バイトで64Kバイト全メモリを指定する。)

たとえば、メモリの位置$3E32_H$への無条件ジャンプは次のようになる。

アドレスA	C3	OPコード
A+1	32	下位アドレス
A+2	3E	上位アドレス

相対ジャンプ命令は2バイト命令であり、2バイト目はその時点でのPCの内容に加えるためのディスプレイスメント（2の補数値）である。

このディスプレイスメントによりOPコードのあるアドレスから＋129～－126の範囲を指定できる。

レジスタ間接ジャンプには3種あって、レジスタ・ペアHL、インデックス・レジスタIX、IYの内容を直接PCにロードして行う。このモードでは、まえで計算した値をもとにジャンプ先を決めてジャンプすることができる。

コール命令はジャンプ命令の特殊な形であり、戻り番地（コール命令のあとのバイトのアドレス）をスタックにプッシュしておいてジャンプを実行するものである。リターン命令はコールの逆で、スタックからデータをポップしてきて、それを直接PCにロードしてジャンプする命令である。

コールおよびリターン命令で、サブルーチンあるいは割り込みの処理が、簡単に行える。

Z－80Aファミリ用に、2つの特殊リターン命令が設けられている。RETI（マスク可能な割り込みからのリターン）とRETN（ノン・マスカブル割り込みからのリターン）の2種で、CPU内ではOPコード$C9_H$の無条件リターンと同様に扱われるが、次の点で異なっている。

RETIは割り込み処理ルーチンの終わりで使用するが、Z－80Aの周辺チップはこの命令の実行を検知して、ネスティングしていた割り込みに対して適切な処理が行われたことを認識できる。

Z－80Aの周辺デバイスの構成を考慮したこの命令を使用すれば、ネスティングしていた割り込みからの正常リターンの手続きが簡単になる。もし、このような命令がなければ、割り込み処理ルーチンが完了し、割り込みをかけてきたデバイスにそれを伝えるためには、次のようなプログラム・ステップが必要であろう。

```
DI          ; このルーチンが始まるまえにまず割り込みを禁止する（ディセーブル割り込み）。
LD  A, n    ; ⎫
OUT  n, A   ; ⎭ サービス・ルーチン完了をデバイスへ通知する。
RET         ; リターン。
```

Z－80Aでは、2バイトのRETI命令で、さきのような6バイトのステップを置き換えることができるのである。これは割り込みサービスの時間をできる限り短縮しなければならない場合には重要となる。

プログラムでループ制御を簡単にするためにDJNZ命令があり、便利なものである。これは2バイトの条件付相対ジャンプ命令で、しかも命令実行ごとにBレジスタがデクリメントされる。Bレジスタが零になるまで繰り返しジャンプを行い、零になれば次の命令ステップに移る。オペランド（ディスプレイスメント）は2の補数値で与える。次に簡単な例を示しておく。

アドレス	命　令	コメント
N、N+1	LD　B、7	；Bレジスタをカウント7にセット。
N+2	(プログラム・ステップ)	；7回繰り返す内容。
∫		
N+9		
N+10、N+11	DJNZ　−10	；もし、Bレジスタが0でなければ N+12からN+2へ戻る。
N+12	(次のステップ)	；(Bレジスタ)＝0のとき、ここへ。

			条件									
			無条件	キャリ	ノン・キャリ	ゼロ	ノン・ゼロ	パリティ偶数	パリティ奇数	負	正	カウント
JUMP "JP"	拡張イミディエット	nn	C3 n n	DA n n	D2 n n	CA n n	C2 n n	EA n n	E2 n n	FA n n	F2 n n	
JUMP "JR"	相対	PC+e	18 e-2	38 e-2	30 e-2	28 e-2	20 e-2					
JUMP "JP"	レジスタ間接	(HL)	E9									
JUMP "JP"	レジスタ間接	(IX)	DD E9									
JUMP "JP"	レジスタ間接	(IY)	FD E9									
"CALL"	拡張イミディエット	nn	CD n n	DC n n	D4 n n	CC n n	C4 n n	EC n n	E4 n n	FC n n	F4 n n	
DECREMENT B, JUMP IF NON ZERO "DJNZ"	相対	PC+e										10 e-2
RETURN "RET"	レジスタ間接	(SP) (SP+1)	C9	D8	D0	C8	C0	E8	E0	F8	F0	
RETURN FROM INT "RETI"	レジスタ間接	(SP) (SP+1)	ED 4D									
RETURN FROM NON MASKABLE INT "RETN"	レジスタ間接	(SP) (SP+1)	ED 45									

あるフラグは1つ以上の
目的で用いられる。
詳細は第5章参照。

表4−11　ジャンプ、コール、リターン　グループ

表4−12に8種のリスタート（RST）命令のOPコードおよびニーモニックを示す。この命令は1バイトで8箇所のいずれかをコールできる。

この命令の特長は、よく使用するルーチンをこれで呼べば、プログラム・ステップ数、すなわちメモリ使用量を節減できる点にある。

コール・アドレス	OPコード	
0000_H	C7	"RST 0"
0008_H	CF	"RST 8"
0010_H	D7	"RST 16"
0018_H	DF	"RST 24"
0020_H	E7	"RST 32"
0028_H	EF	"RST 40"
0030_H	F7	"RST 48"
0038_H	FF	"RST 56"

表4－12　　リスタート

入力／出力

Z－80A CPU には表４－13および表４－14に示すように、多種の入出力命令群がある。

入力／出力デバイスのアドレシングには、絶対モードあるいは、C レジスタを使用したレジスタ間接モードがある。

レジスタ間接モードでは、入出力デバイス群と内部レジスタ間のデータの直接転送が可能である点に注意されたい。さらに、８つのブロック転送命令が設けられている。

これらの命令群はメモリ・ブロック転送と同種のものであるが、HL レジスタ・ペアをメモリ・ソース（出力要求の場合）またはメモリ・デスティネーション（入力要求の場合）のポインタとして用いる点が異なる。一方、B レジスタは同様にバイト・カウンタとして使用する。

C レジスタは入力／出力の必要なデバイスのポートのアドレスを保持する。

B レジスタは８ビット長なので、入出力ブロック転送は256バイトまでである。

次のような入出力命令では n はアドレス・バスの下位８ビットに出力される。

```
IN   A, n
OUT  n, A
```

同時にアキュムレータの内容がアドレス・バスの上位８ビットへ出力される。

入出力ブロック転送も含めたレジスタ間接入出力命令のいずれにおいても、レジスタCの内容は、アドレス・バスの下位８ビットで転送され同時にレジスタBの内容がアドレス・バスの上位８ビットに出力される。

			ソース 入力ポート	
			イミディエット	レジスタ間接
			(n)	(C)
INPUT "IN"	レジスタ・アドレシング	A	DB n	ED 78
		B		ED 40
		C		ED 48
		D		ED 50
		E		ED 58
		H		ED 60
		L		ED 68
"INI"−INPUT & Inc HL, Dec B	レジスタ間接	(HL)		ED A2
"INIR"−INP, Inc HL, Dec B, REPEAT IF B≠0				ED B2
"IND"−INPUT & Dec HL, Dec B				ED AA
"INDR"−INPUT, Dec HL, Dec B, REPEAT IF B≠0				ED BA

デスティネーション入力

ブロック入力コマンド

表4−13　入力命令群

			ソース							
			レジスタ							レジスタ間接
			A	B	C	D	E	H	L	(HL)
"OUT"	イミディエット	(n)	D3 n							
	レジスタ間接	(C)	ED 79	ED 41	ED 49	ED 51	ED 59	ED 61	ED 69	
"OUTI"−OUTPUT Inc HL, Dec b	レジスタ間接	(C)								ED A3
"OTIR"−OUTPUT, Inc HL, Dec B, REPEAT IF B≠0	レジスタ間接	(C)								ED B3
"OUTD"−OUTPUT Dec HL & B	レジスタ間接	(C)								ED AB
"OTDR"−OUTPUT, Dec HL & B, REPEAT IF B≠0	レジスタ間接	(C)								ED BB

ブロック出力コマンド

デスティネーション出力ポート

表4−14　出力命令群

CPU制御命令

表4－15に6つの汎用CPU制御命令を示す。NOPは何もしない命令である。HALTはCPUの動作を次に割り込みが入るまで停止させる命令であるが、この割り込みはDI、EI命令によってソフト的に受け付け可否の制御ができる。

3種類の割り込み動作モードの指定命令によって、CPUはそれぞれ次のような対応する割り込み応答モードになる。

モード0にセットされた場合、割り込みを発しているデバイスがデータ・バス上に何らかの命令を乗せて、CPUがそれを実行する形式となる。

モード1は簡単なもので、CPUは自動的に位置0038_Hからリスタート（RST）するので、外部回路からの指令は何ら必要でない。(前のPCの内容はスタックにプッシュされる。)

モード2は最も強力なもので、全メモリの位置への間接コールができる。このモードでは、CPUはレジスタIの内容により上位8ビットを指定し、割り込みを発生しているデバイスからのデータで下位8ビットを指定し、計16ビット・メモリ・アドレスを形成する。

このアドレスとそれに続くアドレスの内容2バイトにより、サービス・ルーチンの先頭アドレスを指定する。

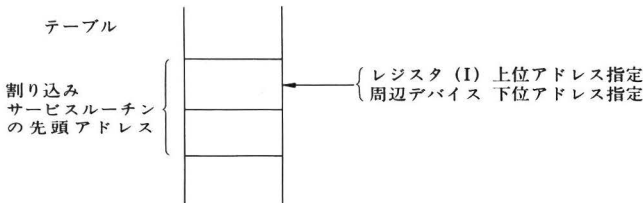
CPUは自動的にこのデータ2バイトで指定されるアドレスへのCALL命令を実行する。

命令	コード	備考
"NOP"	00	
"HALT"	76	
DISABLE INT "(DI)"	F3	
ENABLE INT "(EI)"	FB	
SET INT MODE 0 "IM 0"	ED 46	8080Aモード
SET INT MODE 1 "IM 1"	ED 56	38_H番地へのコール命令
SET INT MODE 2 "IM 2"	ED 5E	レジスタIと割り込みデバイスからの8ビットデータを用いた間接コール命令

表4－15　CPU制御命令

5．フラグ類

Z－80A CPUの2組のフラグ・レジスタは種々のCPU操作による結果を6ビットで保持する。

このうち、4つのビットはプログラムによりテストでき、ジャンプの条件として使用できる。これらについて説明する。

1）キャリ・フラグ（C）—アキュムレータの最上位ビットからの桁上がりでセットされる。加算命令でのキャリ、または減算命令でボロー発生の場合などでセットされる。シフト、ローテイト命令ではこのCビットが関係する。

2）ゼロ・フラグ（Z）—演算の結果、アキュムレータに零がロードされたときにセットされる。他の場合リセットされる。

3）サイン・フラグ（S）—このフラグは符号付数値を扱う場合に使用し、演算の結果が負ならばセットされる。符号としてビット7（MSB）を使用するので、アキュムレータのビット7の内容がこのフラグに置かれる。（負の数ならば1が立つ。）

4）パリティ／オーバフロー・フラグ（P/V）—この2つの目的を持ったフラグは論理演算（AND A, Bなど）が行われた場合にはパリティを示し、また符号付2の補数値の算術演算が行われた場合にはオーバフローの有無を示すために用いられる。Z－80Aのオーバフローはアキュムレータ内の2の補数値が＋127を超えるか、－128より小さくなった場合に出される。次のような加算ではオーバフローとなる。

```
  +120=   0111  1000
  +105=   0110  1001
─────────────────────────────────────────
   C=0    1110  0001  = (−31) オーバフロー発生 (エラー)
```

これはエラーであり、オーバフローがあるがキャリは出ないので、エラー表示のためにオーバフロー・フラグがセットされる。

次のような2の補数値の加算では正しい結果が出て、キャリ・フラグがセットされる。この場合、オーバフローではないので、オーバフロー・フラグはセットされない。

```
  − 5=   1111  1011
  −16=   1111  0000
─────────────────────────────────────
   C=1   1110  1011  = (−21) キャリ発生 (正答)
```

キャリは結果の正誤に直接関係しないので、オーバフロー・フラグが特別に必要である点に注意されたい。論理演算（AND、OR、XOR）では、結果のパリティが偶数ならばこのフラグはセットされ、奇数ならばリセットされる。

プログラマが直接に関与する必要のないフラグ・レジスタの残りのビットは、いずれもBCD算術演算用として使用される。

1）ハーフ・キャリ（H）—BCD演算結果の下位4ビットからのキャリ、ボロー用である。命令DAAを用いて10進補正演算を行う場合、このフラグHを用いて、CPUは自動的にパックされた10進数の加算結果または減算結果に対して補正を行い、正しい結果を算出する。

2）サブトラクト・フラグ（N）—BCD演算で、加算と減算とではアルゴリズムが異なっているので、このフラグを見て、先に実行された命令が加算か減算かを判定し、正しくDAA命令を実行する。

このフラグ・レジスタはプログラムによって調べたり操作したりすることができ、フォーマットは次のようになっている。

S	Z	X	H	X	P/V	N	C

Xは未定義

表5－1に各種のCPU命令でどのようにフラグ・ビットが変化するかを示す。各記号は次のような意味を持つ。

● — フラグ　変化なし。
X — フラグ　不定。
0 — リセットされる。
1 — セットされる。
↕ — 各条件（前述）にしたがって、セットあるいはリセットされる。

この表に書かれていないその他の命令はフラグに影響を与えない。

表５－１には、説明の必要があるいく種かの場合が含まれている。Ｚフラグは、ブロック・サーチにおいて、それらの結果、状態を示すのに用いられる。

ブロック・サーチで、ソースとアキュムレータのデータが一致すればＺフラグはセットされる。またパリティ・フラグは、バイト・カウンタ（BCレジスタ・ペア）が零でなければセットされている。パリティ・フラグはブロック転送の際にも同様になっている。

ブロック入出力命令においては、Ｚフラグはバイト・カウンタとしてのＢレジスタの状態を示すために用いられる。入出力ブロック転送が終了したとき、Ｚフラグは零（すなわち、Ｂ＝０）にリセットされる。他方、メモリ間ブロック転送の終了時には、パリティ・フラグがリセットされる。

リフレッシュ時、あるいはＩレジスタの内容がアキュムレータにロードされたとき、割り込みのイネーブル・フリップ・フロップ（IFF）の内容は、パリティ・フラグに退避されるので、CPUの状態はいつでも完全に保存されている。

命　　令	C	Z	P/V	S	N	H	備　　考
ADD A,s; ADC A,s	↕	↕	V	↕	0	↕	8ビット加算、キャリを含む加算
SUB s; SBC A,s, CP s	↕	↕	V	↕	1	↕	8ビット減算、キャリを含む加算、比較
NEG	↕	↕	V	↕	1	↕	符号反転（ニゲイト）
AND s	0	↕	P	↕	0	1	論理演算
OR s; XOR s	0	↕	P	↕	0	0	
INC	●	↕	V	↕	0	↕	8ビット・インクリメント
DEC s	●	↕	V	↕	1	↕	8ビット・デクリメント
ADD dd, ss	↕	●	●	●	0	X	16ビット加算
ADC HL, ss	↕	↕	V	↕	0	X	16ビットキャリを含む加算
SBC HL, ss	↕	↕	V	↕	1	X	16ビットキャリを含む減算
RLA; RLCA; RRA; RRCA	↕	●	●	●	0	0	ローテイト・アキュムレータ
RL s; RLC s; RR s; RRC s SLA s; SRA s; SRL s	↕	↕	P	↕	0	0	ローテイト、シフトS
RLD, RRD	●	↕	P	↕	0	0	ローテイト・ディジット　左、右
DAA	↕	↕	P	↕	●	↕	デシマル・アジャスト・アキュムレータ
CPL	●	●	●	●	1	1	アキュムレータ補数変換
SCF	1	●	●	●	0	0	セット・キャリ
CCF	↕	●	●	●	0	X	キャリ補数変換
IN r, (C)	●	↕	P	↕	0	0	レジスタ間接入力
INI; IND; OUTI; OUTD	●	↕	X	X	1	X	ブロック入出力
INIR; INDR; OTIR; OTDR	●	1	X	X	1	X	B≠0ならばZ＝0、その他はZ＝1
LDI, LDD	●	X	↕	X	0	0	ブロック転送
LDIR, LDDR	●	X	0	X	0	0	B≠0ならばP/V＝0、その他はP/V＝0
CPI, CPIR, CPD, CPDR	●	↕	↕	X	1	X	ブロック・サーチ A＝(HL)ならばZ＝1、その他はZ＝0 BC≠0ならばP/V＝1、その他はP/V＝0
LDA, I; LD A, R	●	↕	IFF	↕	0	0	IFFの内容がP/Vにコピーされる。
BIT b, s	●	↕	X	X	0	1	Sのビットbの内容がZにコピーされる。

記号の説明

C　：キャリ／リンク・フラグ	結果のMSBからのキャリがあれば、C＝1。
Z　：ゼロ・フラグ	零ならば、Z＝1。
S　：サイン・フラグ	結果のMSBが1ならば、S＝1。
P/V：パリティとオーバフロー兼用フラグ	結果が奇数、またはオーバフローならば、P/V＝1。 結果が偶数ならば、P/V＝0。
H　：ハーフ・キャリフラグ	結果にキャリ、ボローがあれば、H＝1。
N　：加算／減算フラグ	さきの演算が減算ならば、N＝1。
↕　：	操作の結果、変化する。
●　：	操作の結果、変化しない。
0　：	操作により、リセットされる。
1　：	操作により、セットされる。
X　：	無視してよい。
V　：	オーバ・フラグとして扱われる。
P　：	パリティ・フラグとして扱われる。
s　：	8ビット・ロケーション。
R　：	リフレッシュ・カウンタ。
I　：	Iレジスタ（割り込みベクトルの上位バイト用）。
r　：	CPUレジスタ A, B, C, D, E, H, L。
ss　：	16ビット・ロケーション。
n　：	8ビット値（0〜255）。
nn　：	16ビット値（0〜65535）。

表5－1

6. オペコードとタイミング表

この章で、Z－80Aの命令セットをまとめておく。これらをグループ別に表6－1から表6－11に示し、各表で、次の各項について示す。

アセンブリ語ニーモニックのOPコード
マシン語のOPコード
シンボリック・オペレーション（CPUの動作、または操作）
命令実行後のフラグ・レジスタの内容
命令のバイト数（メモリ・サイクル数）
フェッチ、実行に必要なTステート（外部クロック周期）の総数

これらの表はできるだけその表だけで意味がわかるよう配慮されている。

ニーモニックコード	シンボリックオペレーション	フラグ C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント
LD r,r′	r←r′	●	●	●	●	●	●	01 r r′	1	1	4	
LD r,n	r←n	●	●	●	●	●	●	00 r 110 ← n →	2	2	7	
LD r,(HL)	r←(HL)	●	●	●	●	●	●	01 r 110	1	2	7	
LD r,(IX+d)	r←(IX+d)	●	●	●	●	●	●	11 011 101 01 r 110 ← d →	3	5	19	
LD r,(IY+d)	r←(IY+d)	●	●	●	●	●	●	11 111 101 01 r 110 ← d →	3	5	19	
LD (HL),r	(HL)←r	●	●	●	●	●	●	01 110 r	1	2	7	
LD (IX+d),r	(IX+d)←r	●	●	●	●	●	●	11 011 101 01 110 r ← d →	3	5	19	
LD (IY+d),r	(IY+d)←r	●	●	●	●	●	●	11 111 101 01 110 r ← d →	3	5	19	
LD (HL),n	(HL)←n	●	●	●	●	●	●	00 110 110 ← n →	2	3	10	
LD (IX+d),n	(IX+d)←n	●	●	●	●	●	●	11 011 101 00 110 110 ← d → ← n →	4	5	19	
LD (IY+d),n	(IY+d)←n	●	●	●	●	●	●	11 111 101 00 110 110 ← d → ← n →	4	5	19	
LD A,(BC)	A←(BC)	●	●	●	●	●	●	00 001 010	1	2	7	
LD A,(DE)	A←(DE)	●	●	●	●	●	●	00 011 010	1	2	7	
LD A,(nn)	A←(nn)	●	●	●	●	●	●	00 111 010 ← n → ← n →	3	4	13	
LD (BC),A	(BC)←A	●	●	●	●	●	●	00 000 010	1	2	7	
LD (DE),A	(DE)←A	●	●	●	●	●	●	00 010 010	1	2	7	
LD (nn),A	(nn)←A	●	●	●	●	●	●	00 110 010 ← n → ← n →	3	4	13	
LD A,I	A←I	●	↕	IFF2	↕	0	0	11 101 101 01 010 111	2	2	9	
LD A,R	A←R	●	↕	IFF2	↕	0	0	11 101 101 01 011 111	2	2	9	
LD I,A	I←A	●	●	●	●	●	●	11 101 101 01 000 111	2	2	9	
LD R,A	R←A	●	●	●	●	●	●	11 101 101 01 001 111	2	2	9	

r,r′	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

注）r、r′はA、B、C、D、E、H、Lレジスタを指す。
IFF（割り込みイネーブル・フリップ）はP/Vフラグにコピーされる。

フラグ：・＝変化せず、0＝リセット、1＝セット、X＝不定。
↕＝操作の結果でセットまたはリセットされる。

表6－1　8ビット・ロード

ニーモニックコード	シンボリックオペレーション	フラグ						オペコード	バイト数	Mサイクル数	Tステート数	コメント
		C	Z	P/V	S	N	H	76 543 210				
LD dd,nn	dd←nn	●	●	●	●	●	●	00 dd0 001 ← n → ← n →	3	3	10	dd 00 BC 01 DE 10 HL 11 SP
LD IX,nn	IX←nn	●	●	●	●	●	●	11 011 101 00 100 001 ← n → ← n →	4	4	14	
LD IY,nn	IY←nn	●	●	●	●	●	●	11 111 101 00 100 001 ← n → ← n →	4	4	14	
LD HL,(nn)	H←(nn+1) L←(nn)	●	●	●	●	●	●	00 101 010 ← n → ← n →	3	5	16	nnは2バイトの数、下位1バイトはオペコードの直後。 上位1バイトはその次に入る。
LD dd,(nn)	dd_H←(nn+1) dd_L←(nn)	●	●	●	●	●	●	11 101 101 01 dd1 011 ← n → ← n →	4	6	20	
LD IX,(nn)	IX_H←(nn+1) IX_L←(nn)	●	●	●	●	●	●	11 011 101 00 101 010 ← n → ← n →	4	6	20	
LD IY,(nn)	IY_H←(nn+1) IY_L←(nn)	●	●	●	●	●	●	11 111 101 00 101 010 ← n → ← n →	4	6	20	
LD (nn),HL	(nn+1)←H (nn)←L	●	●	●	●	●	●	00 100 010 ← n → ← n →	3	5	16	
LD (nn),dd	(nn+1)←dd_H (nn)←dd_L	●	●	●	●	●	●	11 101 101 01 dd0 011 ← n → ← n →	4	6	20	
LD (nn),IX	(nn+1)←IX_H (nn)←IX_L	●	●	●	●	●	●	11 011 101 00 100 010 ← n → ← n →	4	6	20	
LD (nn),IY	(nn+1)←IY_H (nn)←IY_L	●	●	●	●	●	●	11 111 101 00 100 010 ← n → ← n →	4	6	20	
LD SP,HL	SP←HL	●	●	●	●	●	●	11 111 001	1	1	6	
LD SP,IX	SP←IX	●	●	●	●	●	●	11 011 101 11 111 001	2	2	10	
LD SP,IY	SP←IY	●	●	●	●	●	●	11 111 101 11 111 001	2	2	10	

ニーモニックコード	シンボリックオペレーション	フラグ C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント
PUSH qq	$(SP-2) \leftarrow qq_L$ $(SP-1) \leftarrow qq_H$	●	●	●	●	●	●	11 qq0 101	1	3	11	qq / レジスタ 00 BC 01 DE 10 HL 11 AF
PUSH IX	$(SP-2) \leftarrow IX_L$ $(SP-1) \leftarrow IX_H$	●	●	●	●	●	●	11 011 101 11 100 101	2	4	15	
PUSH IY	$(SP-2) \leftarrow IY_L$ $(SP-1) \leftarrow IY_H$	●	●	●	●	●	●	11 111 101 11 100 101	2	4	15	
POP qq	$qq_H \leftarrow (SP+1)$ $qq_L \leftarrow (SP)$	●	●	●	●	●	●	11 qq0 001	1	3	10	
POP IX	$IX_H \leftarrow (SP+1)$ $IX_L \leftarrow (SP)$	●	●	●	●	●	●	11 011 101 11 100 001	2	4	14	
POP IY	$IY_H \leftarrow (SP+1)$ $IY_L \leftarrow (SP)$	●	●	●	●	●	●	11 111 101 11 100 001	2	4	14	

注) ddはレジスタ・ペアBC、HL、SP。
qqはレジスタ・ペアAF、BC、DE、HL。
添字H、Lはそれぞれ高位バイト、低位バイトを表わす。
例) $BC_L = C$、 $AF_H = A$

フラグ：・＝変化せず。
0＝リセットされる。
1＝セットされる。
X＝不定である。
↕＝操作の結果により、セットまたはリセットされる。

表6－2　16ビット・ロード

ニーモニックコード	シンボリックオペレーション	フラグ C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント
EX DE,HL	DE↔HL	•	•	•	•	•	•	11 101 011	1	1	4	
EX AF,AF′	AF↔AF′	•	•	•	•	•	•	00 001 000	1	1	4	
EXX	(BC DE HL)↔(BC′ DE′ HL′)	•	•	•	•	•	•	11 011 001	1	1	4	レジスタ・ペアの内容とその補助レジスタ・ペアの内容を交換する
EX (SP),HL	H↔(SP+1) L↔(SP)	•	•	•	•	•	•	11 100 011	1	5	19	
EX (SP),IX	IX_H↔(SP+1) IX_L↔(SP)	•	•	•	•	•	•	11 011 101 11 100 011	2	6	23	
EX (SP),IY	IY_H↔(SP+1) IY_L↔(SP)	•	•	•	•	•	•	11 111 101 11 100 011	2	6	23	
LDI	(DE)←(HL) DE←DE+1 HL←HL+1 BC←BC−1	•	•	①↕	•	0	0	11 101 101 10 100 000	2	4	16	ポインタ　1増 バイトカウンタ　1減
LDIR	(DE)←(HL) DE←DE+1 HL←HL+1 BC←BC−1 BC=0まで繰り返す	•	•	0	•	0	0	11 101 101 10 110 000	2 2	5 4	21 16	BC≠0の場合 BC=0の場合
LDD	(DE)←(HL) DE←DE−1 HL←HL−1 BC←BC−1	•	•	①↕	•	0	0	11 101 101 10 101 000	2	4	16	
LDDR	(DE)←(HL) DE←DE-1 HL←HL-1 BC←BC-1 BC=0まで繰り返す	•	•	0	•	0	0	11 101 101 10 111 000	2 2	5 4	21 16	BC≠0の場合 BC=0の場合
CPI	A−(HL) HL←HL+1 BC←BC−1	•	②↕	①↕	↕	1	↕	11 101 101 10 100 001	2	4	16	
CPIR	A−(HL) HL←HL+1 BC←BC−1 A=(HL)またはBC=0まで繰り返す	•	②↕	①↕	↕	1	↕	11 101 101 10 110 001	2 2	5 4	21 16	BC≠0かつA≠(HL)の場合 BC=0またはA=(HL)の場合
CPD	A−(HL) HL←HL−1 BC←BC−1	•	②↕	①↕	↕	1	↕	11 101 101 10 101 001	2	4	16	
CPDR	A−(HL) HL←HL−1 BC←BC−1 A=(HL)またはBC=0まで繰り返す	•	②↕	①↕	↕	1	↕	11 101 101 10 111 001	2 2	5 4	21 16	BC≠0かつA≠(HL)の場合 BC=0またはA=(HL)の場合

注）①BC－１＝０ならばP/Vは０、その他は１。
②A＝(HL)ならばZは１、その他は０。

フラグ：・＝変化せず。
０＝セット、１＝リセット。
↕＝操作結果で変化する。

表6－3　交換、ブロック転送、サーチ

ニーモニックコード	シンボリックオペレーション	C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント	
ADD A,r	A←A+r	↕	↕	V	↕	0	↕	10 [000] r	1	1	4	r	レジスタ
ADD A,n	A←A+n	↕	↕	V	↕	0	↕	11 [000] 110	2	2	7	000	B
								← n →				001	C
ADD A,(HL)	A←A+(HL)	↕	↕	V	↕	0	↕	10 [000] 110	1	2	7	010	D
ADD A,(IX+d)	A←A+(IX+d)	↕	↕	V	↕	0	↕	11 011 101	3	5	19	011	E
								10 [000] 110				100	H
								← d →				101	L
ADD A,(IY+d)	A←A+(IY+d)	↕	↕	V	↕	0	↕	11 111 101	3	5	19	111	A
								10 [000] 110					
								← d →					
ADC A,s	A←A+s+CY	↕	↕	V	↕	0	↕	[001]				sは，ADD命令と同様にr, n, (HL), (IX+d), (IY+d) のいずれかを示す。ワクで囲ったビットを上のADD命令の[000]のかわりに置く。	
SUB s	A←A−s	↕	↕	V	↕	1	↕	[010]					
SBC A,s	A←A−s−CY	↕	↕	V	↕	1	↕	[011]					
AND s	A←A∧s	0	↕	P	↕	0	1	[100]					
OR s	A←A∨s	0	↕	P	↕	0	0	[110]					
XOR s	A←A∀s	0	↕	P	↕	0	0	[101]					
CP s	A—s	↕	↕	V	↕	1	↕	[111]					
INC r	r←r+1	●	↕	V	↕	0	↕	00 r [100]	1	1	4		
INC (HL)	(HL)←(HL)+1	●	↕	V	↕	0	↕	00 110 [100]	1	3	11		
INC (IX+d)	(IX+d)← (IX+d)+1	●	↕	V	↕	0	↕	11 011 101	3	6	23		
								00 110 [100]					
								← d →					
INC (IY+d)	(IY+d)← (IY+d)+1	●	↕	V	↕	0	↕	11 111 101	3	6	23		
								00 110 [100]					
								← d →					
DEC m	m←m−1	●	↕	V	↕	1	↕	[101]				mはINC命令と同様にr, (HL) (IX+d), (IY+d) のいずれかを示す。オペコードはINC命令の[100]を[101]に変更したものに同じである。	

注）Vはオーバフロー・フラグとして、Pはパリティフラグとして扱われることを意味する。

フラグ：・＝変化せず。
0＝リセットされる。
1＝セットされる。
X＝不定。
↕＝操作結果により、セットまたはリセットされる。

表6－4　8ビット算術・論理演算

ニーモニックコード	シンボリックオペレーション	フラグ						オペコード	バイト数	Mサイクル数	Tステート数	コメント
		C	Z	P/V	S	N	H	76 543 210				
DAA		↕	↕	P	↕	●	↕	00 100 111	1	1	4	2進化10進数(BCD)補正
CPL	A←Ā	●	●	●	●	1	1	00 101 111	1	1	4	1の補数　Acc
NEG	A←Ā+1	↕	↕	V	↕	1	↕	11 101 101 01 000 100	2	2	8	2の補数　Acc
CCF	CY←$\overline{CY}$	↕	●	●	●	0	X	00 111 111	1	1	4	キャリの反転
SCF	CY←1	1	●	●	●	0	0	00 110 111	1	1	4	キャリ・セット
NOP	何も実行しないが PC←PC+1	●	●	●	●	●	●	00 000 000	1	1	4	
HALT	CPU停止	●	●	●	●	●	●	01 110 110	1	1	4	
DI	IFF←0	●	●	●	●	●	●	11 110 011	1	1	4	ディセーブル割り込み
EI	IFF←1	●	●	●	●	●	●	11 111 011	1	1	4	イネーブル割り込み
IM 0	割り込みモード 0にする	●	●	●	●	●	●	11 101 101 01 000 110	2	2	8	割り込みモードのセット
IM 1	割り込みモード 1にする	●	●	●	●	●	●	11 101 101 01 010 110	2	2	8	
IM 2	割り込みモード 2にする	●	●	●	●	●	●	11 101 101 01 011 110	2	2	8	

注）IFFは割り込みフリップ・フロップ。
　　CYはキャリ・フリップ・フロップ。

フラグ：・＝変化せず。
　　0＝リセットされる。
　　1＝セットされる。
　　X＝不定。
　　↕＝操作の結果により、セットまたはリセットされる。

表6－5　各種操作およびCPU制御

ニーモニックコード	シンボリックオペレーション	フラグ						オペコード	バイト数	Mサイクル数	Tステート数	コメント
		C	Z	P/V	S	N	H	76 543 210				
ADD HL,ss	HL←HL+ss	↕	●	●	●	0	X	00 ss1 001	1	3	11	ss｜レジスタ 00｜BC 01｜DE 10｜HL 11｜SP
ADC HL,ss	HL←HL+ss+CY	↕	↕	V	↕	0	X	11 101 101 01 ss1 010	2	4	15	
SBC HL,ss	HL←HL−ss−CY	↕	↕	V	↕	1	X	11 101 101 01 ss0 010	2	4	15	
ADD IX,pp	IX←IX+pp	↕	●	●	●	0	X	11 011 101 00 pp1 001	2	4	15	pp｜レジスタ 00｜BC 01｜DE 10｜IX 11｜SP
ADD IY,rr	IY←IY+rr	↕	●	●	●	0	X	11 111 101 00 rr1 001	2	4	15	rr｜レジスタ 00｜BC 01｜DE 10｜IY 11｜SP
INC ss	ss←ss+1	●	●	●	●	●	●	00 ss0 011	1	1	6	
INC IX	IX←IX+1	●	●	●	●	●	●	11 011 101 00 100 011	2	2	10	
INC IY	IY←IY+I	●	●	●	●	●	●	11 111 101 00 100 011	2	2	10	
DEC ss	ss←ss−1	●	●	●	●	●	●	00 ss1 011	1	1	6	
DEC IX	IX←IX−1	●	●	●	●	●	●	11 011 101 00 101 011	2	2	10	
DEC IY	IY←IY−1	●	●	●	●	●	●	11 111 101 00 101 011	2	2	10	

注）ssはレジスタ・ペアBC、DE、HL、SP。
ppはレジスタ・ペアBC、DE、IX、SP。
rrはレジスタ・ペアBC、DE、IY、SP。

フラグ：・＝変化せず。
0＝リセットされる。
X＝不定。
↕＝操作の結果により、セットまたはリセットされる。

表6−6　16ビット算術演算

ニーモニックコード	シンボリックオペレーション	C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント
RLC A	CY ← A[7 ← 0]	↕	●	●	●	0	0	00 000 111	1	1	4	アキュムレータの内容を左へローテイトする
RL A	CY ← A[7 ← 0]	↕	●	●	●	0	0	00 010 111	1	1	4	
RRC A	A[7 → 0] → CY	↕	●	●	●	0	0	00 001 111	1	1	4	アキュムレータの内容を右へローテイトする
RR A	A[7 → 0] → CY	↕	●	●	●	0	0	00 011 111	1	1	4	
RLC r	CY ← s[7 ← 0]	↕	↕	P	↕	0	0	11 001 011 00 [000] r	2	2	8	レジスタrの内容を左へローテイトする
RLC (HL)		↕	↕	P	↕	0	0	11 001 011 00 [000] 110	2	4	15	
RLC (IX+d)		↕	↕	P	↕	0	0	11 011 101 11 001 011 ← d → 00 [000] 110	4	6	23	
RLC (IY+d)		↕	↕	P	↕	0	0	11 111 101 11 001 011 ← d → 00 [000] 110	4	6	23	
RL s	CY ← s[7 ← 0]	↕	↕	P	↕	0	0	[010]				オペランドsにはr,(HL),(IX+d),(IY+d)が用いられる
RRC s	s[7 → 0] → CY	↕	↕	P	↕	0	0	[001]				
RR s	s[7 → 0] → CY	↕	↕	P	↕	0	0	[011]				
SLA s	CY ← s[7 ← 0] ← 0	↕	↕	P	↕	0	0	[100]				
SRA s	s[7 → 0] → CY	↕	↕	P	↕	0	0	[101]				
SRL s	0 → s[7 → 0] → CY	↕	↕	P	↕	0	0	[111]				
RLD	A[7 4 3 0] (HL)[7 4 3 0]	●	↕	P	↕	0	0	11 101 101 01 101 111	2	5	18	
RRD	A[7 4 3 0] (HL)[7 4 3 0]	●	↕	P	↕	0	0	11 101 101 01 100 111	2	5	18	

r	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

フラグ：・＝変化せず。
0＝リセットされる。
1＝セットされる。
X＝不定。
↕＝操作結果により、セットまたはリセットされる。

表6－7　ローテイト、シフト

ニーモニックコード	シンボリックオペレーション	フラグ C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント
BIT b, r	Z←$\overline{r}_b$	●	↕	X	X	0	1	11 001 011 01 b r	2	2	8	
BIT b, (HL)	Z←$\overline{(HL)}_b$	●	↕	X	X	0	1	11 001 011 01 b 110	2	3	12	
BIT b, (IX+d)	Z←$\overline{(IX+d)}_b$	●	↕	X	X	0	1	11 011 101 11 001 011 ← d → 01 b 110	4	5	20	
BIT b, (IY+d)	Z←$\overline{(IY+d)}_b$	●	↕	X	X	0	1	11 111 101 11 001 011 ← d → 01 b 110	4	5	20	
SET b, r	r_b←1	●	●	●	●	●	●	11 001 011 [11] b r	2	2	8	
SET b, (HL)	$(HL)_b$←1	●	●	●	●	●	●	11 001 011 [11] b 110	2	4	15	
SET b, (IX+d)	$(IX+d)_b$←1	●	●	●	●	●	●	11 011 101 11 001 011 ← d → [11] b 110	4	6	23	
SET b, (IY+d)	$(IY+d)_b$←1	●	●	●	●	●	●	11 111 101 11 001 011 ← d → [11] b 110	4	6	23	
RES b, s	s_b←0 s≡r, (HL), (IX+d), (IY+d)							[10]				オペランドsのビットbをリセットする

r	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

b	テストビット
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

注）m_bの$_b$はmの示すメモリの位置またはレジスタのビット0～7を示す。
フラグ：・＝変化せず。
0＝リセットされる。
1＝セットされる。
X＝不定。
↕＝操作の結果により、セットまたはリセットされる。

表6－8　ビット操作、テスト

ニーモニックコード	シンボリックオペレーション	フラグ C	Z	P/V	S	N	H	オペコード 76 543 210	バイト数	Mサイクル数	Tステート数	コメント
JP nn	PC←nn	●	●	●	●	●	●	11 000 011 ← n → ← n →	3	3	10	
JP cc,nn	ccが真なら PC←nn 偽なら次の命令へ	●	●	●	●	●	●	11 cc 010 ← n → ← n →	3	3	10	cc　条件 000　NZnon zero 001　Z zero 010　NCnon carry 011　C carry 100　PO parity odd 101　PE parity even 110　P sign positive 111　M sign negative
JR e	PC←PC+e	●	●	●	●	●	●	00 011 000 ← e-2 →	2	3	12	
JR C,e	C=0なら次の命令へ	●	●	●	●	●	●	00 111 000 ← e-2 →	2	2	7	
	C=1なら PC←PC+e								2	3	12	
JR NC,e	C=1なら 次の命令へ	●	●	●	●	●	●	00 110 000 ← e-2 →	2	2	7	
	C=0なら PC←PC+e								2	3	12	
JR Z,e	Z=0なら次の命令へ	●	●	●	●	●	●	00 101 000 ← e-2 →	2	2	7	
	Z=1なら PC←PC+e								2	3	12	
JR NZ,e	Z=1なら次の命令へ	●	●	●	●	●	●	00 100 000 ← e-2 →	2	2	7	
	Z=0なら PC←PC+e								2	3	12	
JP (HL)	PC←HL	●	●	●	●	●	●	11 101 001	1	1	4	
JP (IX)	PC←IX	●	●	●	●	●	●	11 011 101 11 101 001	2	2	8	
JP (IY)	PC←IY	●	●	●	●	●	●	11 111 101 11 101 001	2	2	8	
DJNZ,e	B←B−1 B=0なら次の命令へ	●	●	●	●	●	●	00 010 000 ← e-2 →	2	2	8	B=0の場合
	B≠0なら PC←PC+e								2	3	13	B≠0の場合

注) eは相対アドレシング・モードでのイクステンション。
eは符号付2の補数値（−126～+129)。
e−2はPCが自動的に+2されてしまうので、これをキャンセルした値である。
eそのものは、OPコードの位置から計算した値である。

フラグ： ・=変化せず。
0=リセットされる。
1=セットされる。
X=不定。
↕=操作の結果により、セットまたはリセットされる。

表6−9　ジャンプ

ニーモニックコード	シンボリックオペレーション	フラグ						オペコード	バイト数	Mサイクル数	Tステート数	コメント
		C	Z	P/V	S	N	H	76 543 210				
CALL nn	$(SP-1) \leftarrow PC_H$	●	●	●	●	●	●	11 001 101	3	5	17	
	$(SP-2) \leftarrow PC_L$							← n →				
	$PC \leftarrow nn$							← n →				
CALL cc,nn	ccが真ならば	●	●	●	●	●	●	11 cc 100	3	3	10	ccが偽の場合
	CALL nnと同じ、							← n →				
	偽なら次の命令へ							← n →	3	5	17	ccが真の場合
RET	$PC_L \leftarrow (SP)$	●	●	●	●	●	●	11 001 001	1	3	10	
	$PC_H \leftarrow (SP+1)$											
RET cc	ccが真ならばRET	●	●	●	●	●	●	11 cc 000	1	1	5	ccが偽の場合
	と同じ、偽なら次								1	3	11	ccが真の場合
	の命令へ											
RETI	割り込みからのリ	●	●	●	●	●	●	11 101 101	2	4	14	
	ターン							01 001 101				
RETN	NMIからのリター	●	●	●	●	●	●	11 101 101	2	4	14	
	ン							01 000 101				
RST p	$(SP-1) \leftarrow PC_H$	●	●	●	●	●	●	11 t 111	1	3	11	
	$(SP-2) \leftarrow PC_L$											
	$PC_H \leftarrow 0$											
	$PC_L \leftarrow P$											

cc	条件
000	NZ non zero
001	Z zero
010	NC non carry
011	C carry
100	PO parity odd
101	PE parity even
110	P sign positive
111	M sign negative

t	P
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

フラグ：・＝変化せず。
0＝リセットされる。
1＝セットされる。
X＝不定。
↕＝操作の結果により、セットまたはリセットされる。

表6－10　コール、リターン

ニーモニックコード	シンボリックオペレーション	フラグ						オペコード	バイト数	Mサイクル数	Tステート数	コメント
		C	Z	P/V	S	N	H	76 543 210				
IN A,(n)	A←(n)	●	●	●	●	●	●	11 011 011 ← n →	2	3	11	n to A_0~A Acc to A_8~A_{15}
IN r,(C)	r←(C) r=110の場合は、 フラグのみ影響を 受ける	●	↕ ①	P	↕	0	0	11 101 101 01 r 000	2	3	12	C to A_0~A_7 B to A_8~A_{15}
INI	(HL)←(C) B←B−1 HL←HL+1	●	↕	X	X	1	X	11 101 101 10 100 010	2	4	16	C to A_0~A_7 B to A_8~A_{15}
INIR	(HL)←(C) B←B−1 HL←HL+1 B=0まで繰り返す	●	1 ①	X	X	1	X	11 101 101 10 110 010	2 2	5 (B≠0のとき) 4 (B=0のとき)	21 16	C to A_0~A_7 B to A_8~A_{15}
IND	(HL)←(C) B←B−1 HL←HL−1	●	↕	X	X	1	X	11 101 101 10 101 010	2	4	16	C to A_0~A_7 B to A_8~A_{15}
INDR	(HL)←(C) B←B−1 HL←HL−1 B=0まで繰り返す	●	1	X	X	1	X	11 101 101 10 111 010	2 2	5 (B≠0のとき) 4 (B=0のとき)	21 16	C to A_0~A_7 B to A_8~A_{15}
OUT (n),A	(n)←A	●	●	●	●	●	●	11 010 011	2	3	11	n to A_0~A_7 Acc to A_8~A_{15}
OUT (C),r	(C)←r	●	● ①	●	●	●	●	11 101 101 01 r 001	2	3	12	C to A_0~A_7 B to A_8~A_{15}
OUTI	(C)←(HL) B←B−1 HL←HL+1	●	↕	X	X	1	X	11 101 101 10 100 011	2	4	16	C to A_0~A_7 B to A_8~A_{15}
OTIR	(C)←(HL) B←B−1 HL←HL+1 B=0まで繰り返す	●	1 ①	X	X	1	X	11 101 101 10 110 011	2 2	5 (B≠0のとき) 4 (B=0のとき)	21 16	C to A_0~A_7 B to A_8~A_{15}
OUTD	(C)←(HL) B←B−1 HL←HL−1	●	↕	X	X	1	X	11 101 101 10 101 011	2	4	16	C to A_0~A_7 B to A_8~A_{15}
OTDR	(C)←(HL) B←B−1 HL←HL−1 B=0まで繰り返す	●	1	X	X	1	X	11 101 101 10 111 011	2 2	5 (B≠0のとき) 4 (B=0のとき)	21 16	C to A_0~A_7 B to A_8~A_{15}

注) ①B-1が零になればZフラグがセットされ、それ以外のときはリセットされる。
A_{0-15}はアドレス・バス。

フラグ： ・＝変化せず。
0＝リセットされる。
1＝セットされる。
X＝不定。
↕操作結果により、セットまたはリセットされる。

表6－11　入力／出力

7．割り込み応答

割り込みは、周辺デバイスがCPUの通常処理を一時保留させ、周辺デバイスの処理ルーチンを実行させるために設けられている。
普通このサービス（処理）には、CPUと周辺とのデータ、ステータス、あるいは制御情報の送受などが含まれている。
このサービス・ルーチンが終了すれば、CPUは割り込みがかかったまえの処理に戻る。

割り込みのイネーブル／ディセーブル

Z－80A CPUには、ソフト的にマスク可能な割り込みの２つの割り込み入力端子が設けられている~~入力端子が設けられている~~。
ノン・マスカブル割り込み（NMI）はプログラムによってディセーブル（禁止）できず、周辺デバイスが要求を出せば必ず受け付けられる。この割り込みは普通重要な機能、たとえば不意の停電のようないつ起こるかわからない事故に対する処置用として使用される。
マスカブル割り込み（INT）はプログラムによりイネーブル（許可）、ディセーブル（禁止）を自由に選択できるもので、計時（タイマ）のように途中で邪魔（割り込み）が入っては困る場合に、プログラムにより適当にディセーブルしておくことができる。
CPU内にイネーブル・フリップ・フロップ（IFF）が設けられているので、これをプログラムによってEI（イネーブル割り込み）またはDI（ディセーブル割り込み）命令でセット、またはリセットする。
IFFがリセットされれば、CPUは割り込みを受け付けなくなる。

実際には、IFFはIFF$_1$とIFF$_2$の２つのイネーブル・フリップ・フロップにより構成される。

IFF$_1$	IFF$_2$
リセットされれば割り込み禁止となる。	IFF$_1$の一時保持用

IFF$_1$の状態で、イネーブルかディセーブルかが決まる。IFF$_2$はIFF$_1$の一時保持用である。このIFF$_2$が必要な理由をあとで述べる。

CPUがリセットされたとき、IFF$_1$、IFF$_2$もリセット状態となり、割り込みはディセーブルとなる。この後、任意の時点でEI命令を使用して割り込みをイネーブルにすることができる。
EI命令が実行されたとき、待たされている割り込み要求はEI命令の次の命令が実行されたあとでなければ受け付けられない。
この１命令分の遅延は、次の命令がリターン命令の場合、EI命令を実行させたとき、リターン命令が完了した後で待機中の割り込みが受け付けられるようにするために、設けられている。
EI命令でIFF$_1$、IFF$_2$の両方がセットされ、イネーブル状態となる。割り込みが受け付けられたときには自動的に両方ともリセットされ、プログラマが新たにEI命令を実行させることがなければ割り込みは禁止されたままになる。
上に述べた場合においては、いずれもIFF$_1$、IFF$_2$はつねに同様である点に注意されたい。

IFF$_2$はNMI（ノン・マスカブル割り込み）の発生時にIFF$_1$の状態をセーブしておくために設けられている。
NMIが受け付けられたとき、続けて割り込みが入るのを禁止するためにIFF$_1$はリセットされるが、さきの状態がリセットかセットか（つまり、今INTの処理中であるか否か）を保存しておく必要がある。IFF$_2$はこのためのもので、IFF$_1$と同じ状態をIFF$_2$に入れておく。
LD命令（LD　A、I あるいはLD　A、R）を実行したとき、IFF$_2$の状態がパリティ・フラグにコピーされるので、それをテストしたり、別の場所にストアしたりすることができる。

RETN（ノン・マスカブル割り込みからのリターン）命令を実行すると、IFF$_2$の内容がIFF$_1$にコピー・バックされ、NMIの入るまえのIFF$_1$の状態が再生される。
図７－１に、２つのフリップ・フロップの状態を示す。

操 作	IFF_1	IFF_2	
CPUリセット	0	0	
DI	0	0	
EI	1	1	
LD A, I	●	●	IFF_2→パリティ・フラグ
LD A, R	●	●	IFF_2→パリティ・フラグ
NMI	0	●	
RETN	IFF_2	●	IFF_2→IFF_1
DI, then NMI	0	0	
EI, then NMI	0	1	
EI, NMI, then RETN	1	1	●＝変化なしの意味

図7－1 IFF_1, IFF_2の状態

CPUの応答

ノン・マスカブル

ノン・マスカブル割り込みはどの時点でも受け付けられる。これが入力すると、CPUはフェッチした次の命令を無視して、メモリ番地0066_Hにある命令から実行する。つまり、あたかもリスタート命令をフェッチしたかのように動作するが、この場合、正しくはプログラム可能な8つのリスタート・ロケーションのいずれかにリスタートするわけではない。メモリのページ・ゼロの特定のアドレス（66_H）へのコール命令の実行である。

マスカブル

Z－80A CPUにはプログラムで指定できる3種のマスク可能な割り込みモードがある。

モード 0

このモードは8080Aの割り込み応答モードと同様である。このモードでは、割り込みをかけているデバイスがデータ・バス上に何らかの命令を置き、CPUがそれを実行する。

割り込みをかけているデバイスは、プログラム・メモリに代って次に実行すべき命令をデータ・バスに乗せる。この命令は1バイト命令の場合、通常リスタート命令である。3バイトのコール命令を用いた場合、メモリのどの位置からでも実行を始めることができる。

必要なクロック・サイクル数は通常の数より2クロック多くなる。これはCPUが自動的に2つのウエイト・サイクルを挿入して、割り込み制御用の外部デージー・チェーンを全部動作させせるのに必要な時間をとるためである。

割り込み応答の詳細なタイミングについては第4章に示してある。

RESETを行えば、自動的にCPUはモード0になる。

モード 1

このモードをプログラムで選択しておくと、CPUは割り込みに対し、位置0038_Hへのリスタートを実行する。この応答はコールする位置が0066_Hの代りに、0038_Hになっている以外はNMI応答とほぼ同様である。他の異なる点として、2つのウエイト・サイクルが追加される点がある。

モード 2

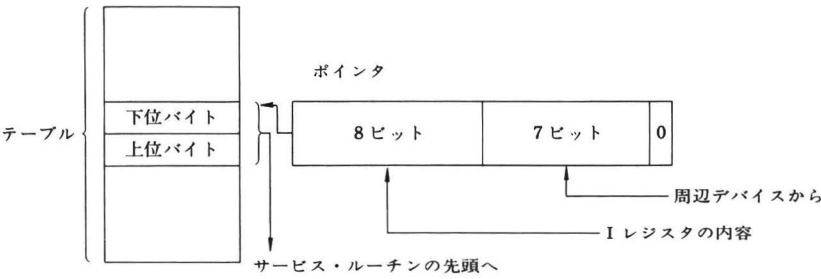
このモードは最も強力な応答モードであり、1バイトの指定でどのメモリの位置でも間接コールができる。

このモードを使用する場合、まえもってプログラムにより割り込みサービス・ルーチンのスタート・アドレス（16ビット）のテーブルを適当な位置に配置しておく。この位置はメモリのどの場所でもよい。

割り込みを受け付けたとき、16ビットのポインタで、必要な割り込みサービス・ルーチンのスタート・アドレスをテーブルからもってくるが、このポインタとして上位8ビットにはIレジスタの内容を充当する。

このIレジスタには、まえもってプログラムによりLD命令で必要な値をロードしておかねばならない。

CPUのリセットによりIレジスタもクリアされるので、Iレジスタの初期値は零である。

ポインタの下位8ビットには割り込みをかけているデバイスから供給しなければならないが、実際は7ビットが有効で最下位ビットはこのモードではつねに零である。というのは、テーブル上のスタート・アドレスは2バイトずつで、この2バイトは必ず偶数アドレスから順に下位バイト、上位バイトと入れておかねばならないからである。

プログラムにより割り込みを受け付けるまでに、必要なアドレスをテーブルにセットしておけばよい。このテーブルを読み出し・書き込みメモリ内に指定しておくと、いつでもプログラムで変更ができるので、異なる周辺デバイスを異なったサービス・ルーチンで処理することもできる。

割り込みをかけているデバイスがポインタの下位8ビットを乗せたあと、CPUは自動的に次に実行すべきPC（プログラム・カウンタ）の内容をスタックにプッシュし、テーブルからスタート・アドレスを持ってそのアドレスへジャンプする。

この応答モードでは、19クロック周期を必要とし、7クロックで割り込みをかけているデバイスからの下位8ビットをフェッチ、6クロックでPCの内容のセーブ、6クロックでジャンプ・アドレスを得る。

Z−80A周辺デバイスはすべてデージー・チェーン構造の割り込み優先順位回路を含んでおり、割り込みアクノリッジ期間中CPUにベクトル（プログラムされている）を自動的に供給するようになっている点に注意されたい。

これについてはZ−80A PIO、Z−80A CTCなどの仕様を参照のこと。

A.2　Z80A-PIOテクニカルデータ

1．緒論

Z－80A PIO（Parallel Input/Output Interface Controller）はZ－80A CPUと周辺装置とのインターフェース用のデバイスで、種々の動作モードをプログラムで指示することができる。また本デバイスはTTLコンパチブルの入出力ポートを2個もっている。

Z－80A PIOはこれをCPUのインターフェース・デバイスとして用いると、余分な外部論理回路を追加しないで広範囲の周辺装置に適用することができる。

Z－80A PIOと接続可能な周辺装置は、たとえば、キーボード、紙テープ・リーダ、紙テープ・パンチャ、プリンタなどがある。

Z－80A PIOはNチャネル・シリコンゲート・デプリーション・ロード技術で製造され、40ピンDIPパッケージである。

Z－80A PIOの主な特長は、

- 2つの独立した8ビット双方向性の周辺装置用インターフェース・ポートがあり、それぞれにデータ転送の制御用の〝ハンドシェーク〞線がある。
- 割り込みは高速応答に適した〝ハンドシェーク〞で起動される。
- 各ポートごとに、次の動作モードの選択ができる。
  - バイト出力
  - バイト入力
  - バイト単位の双方向バス（ポートAのみ）
  - ビット制御モード

  これらにはすべてハンドシェークで制御される割り込み機能がある。
- 外部論理回路なしで自動的に割り込みベクトルを出せるデージー・チェーン構造の割り込み優先順位回路を内蔵している。
- 8出力はダーリントン・トランジスタを駆動できる。
- 全入出力はTTLコンパチブルである。
- 単一5V電源、単相クロックのみで動作する。

Z－80A PIOは他のインターフェース・コントローラと異なり、周辺装置とCPUとの間のデータ転送を割り込み制御の方法により効率よく行うことができる。またZ－80A PIOには、Z－80A CPUの効率の良い割り込み処理能力を入出力データの転送期間に充分に発揮できるように、割り込み処理用の論理回路が組み込まれており、かつ多重の割り込みをネストさせるために必要な論理回路も内蔵している。このため割り込み処理用の余分な外部回路を必要としない。

その他、周辺装置が特定の状態になったときに、割り込み信号が出るようにプログラムしておくこともできるようになっているのも特記すべき点である。たとえば、プログラミングによって、ある周辺装置に特定の状態となる条件が発生したときに割り込みを出すことができる。

この割り込みの能力によってプロセッサが周辺装置のステータスをポーリングする（調べる）間の時間を節約するのに役立たせることができる。

なお、Z－80A PIOは周波数の上限を4 MHzまで保証したZ－80 PIO高速タイプである。

2. アーキテクチャ

図2－1にZ－80A PIOのブロック図を示す。

Z－80A PIOの内部は次の各部で構成されている。

- ●Z－80A CPU用バス・インターフェース
- ●内部制御回路
- ●ポートA入出力論理回路
- ●ポートB入出力論理回路
- ●割り込み制御回路

CPUバス・インターフェース用論理回路は他の外部回路を追加せずに直接CPUに接続できるようになっているが、大きなシステムでは必要に応じてアドレス・デコーダやライン・バッファを付加しなければならないこともある。

内部制御回路はCPUデータ・バスを周辺装置用インターフェース部（ポートA，B）と同期させるためのものである。2つの入出力ポート（A，B）は、ほぼ同様に使用でき、直接周辺装置と結合する部分である。

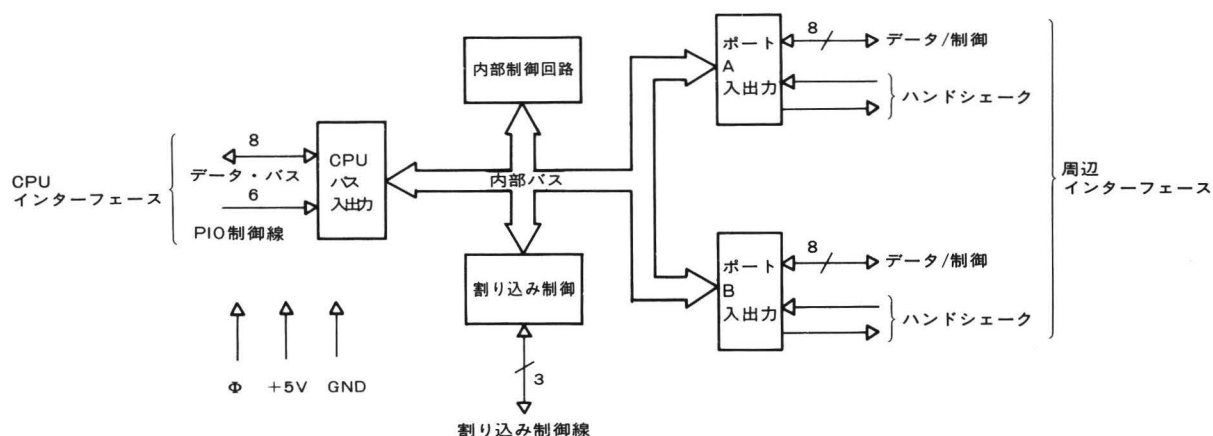



図2－1　　Z－80A PIOブロック図

ポート入出力論理回路は図2－2に示すように、〝ハンドシェーク″用論理回路を含む6つのレジスタから構成されている。

これらはそれぞれ8ビット・データ入力レジスタ、8ビット・データ出力レジスタ、2ビットのモード制御用レジスタ、8ビットのマスク・レジスタ、8ビット入出力選択レジスタ、2ビットのマスク制御用レジスタである。

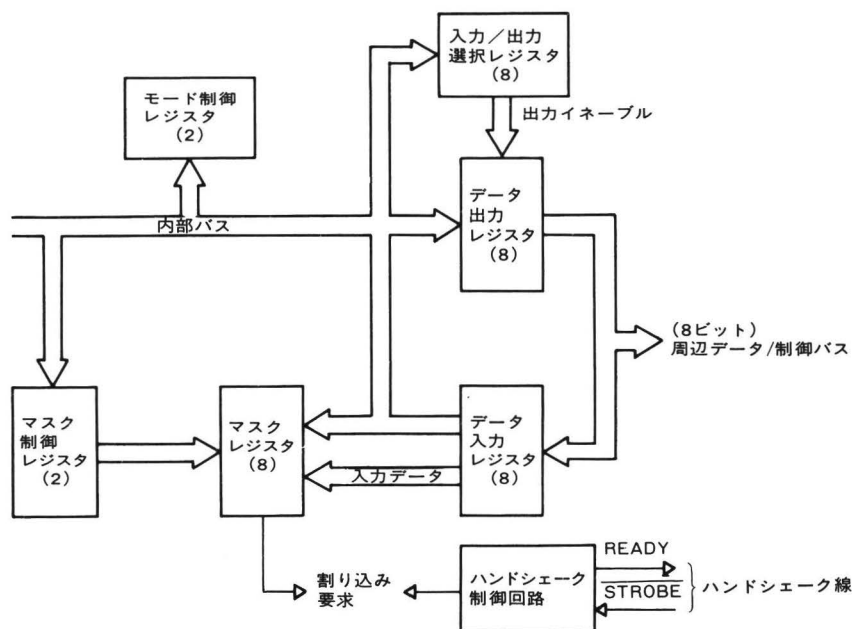



図2－2　　ポート入出力ブロック図

2ビットのモード制御用レジスタは4つの動作モード（バイト入力、バイト出力、バイト双方向バス、ビット制御モード）のうちいずれかを指定するためのもので、CPUからの制御で設定される。

周辺装置とCPUとの間のデータ転送はすべて入出力用レジスタを介して行われ、CPUからの出力データは出力レジスタを通して周辺装置へ送られ、CPUへの入力データは入力レジスタを通してCPUに読み込まれる。データの入出力はどの時点で行ってもよい。

各ポートのハンドシェーク線はPIOと周辺装置との間のデータ転送の制御用として用いる。

8ビット・マスク・レジスタと8ビット入出力選択用レジスタはビット制御モードのときにのみ使用される。このモードでは、周辺と結ばれるデータ転送、もしくは制御用の8本のバス端子を個々に入力用まなは出力用に指定できる。これはプログラムで選択レジスタを設定することにより行う。

このモードでは、特定の割り込み発生条件を設定しておくためにマスク・レジスタを使用する。このマスク・レジスタは入出力用バス端子のどれをモニタするかを決めるために使用し、またどのようにモニタするかは次のようにマスク制御レジスタを使用して行う。

2ビットのマスク制御レジスタは、モニタすべきポート端子の入力すべてについて、AND条件をとるか、OR条件をとるか、あるいは入力が〝H″レベルのときアクティブとするか、〝L″レベルのときアクティブとするかを規定するためにある。これらの指定に合ったステータスになれば、割り込みが発生する。

このように、あらかじめ条件設定をプログラミングしておく方式では、周辺装置の特定のステータスで自動的に割り込みが発生させられるので、CPUが周辺のステータス・チェックをする必要がないという利点がある。システム内に3種の緊急条件がある場合、プログラムでこのうちの1種でも起これば（OR条件）、あるいは3種すべてが起これば（AND条件）割り込みを発生させる、といった使い分けもできる。

割り込み制御用論理回路はCPUの割り込みに関するすべての処理を行い、優先順位の決定などについても関与している部分である。デージー・チェーンの接続によって、各デバイスはその物理的な位置によって優先順位が決まり、CPUに近いデバイスほど優先順位が高くなる。このデージー・チェーンを形成するために、各PIOから2本の線（IEI、IEO）が出ている。また、同一PIO内ではポートAがポートBより優先順位が高い。

バイト入力モード、バイト出力モード、バイト双方向バスモードでは、周辺装置からの次のデータ・バイト転送の要求のあるたびごとに割り込みが発生する。

ビット制御モードでは、周辺のステータスがプログラムした値と一致したとき割り込みが発生する。

PIOはネスト構造の割り込みに関して完全な制御機能をもっていて、CPUが上位の優先順位をもつデバイスの割り込みサービスを行っているときには、下位のデバイスからのサービス要求は通さない（下位の優先順位のデバイスから、割り込みを発生させない）ようになっている。

しかし、上位の優先順位のデバイスは、サービス中のデバイスが下位のものであれば、割り込みをかけることができるようになっている。

モード2で、割り込みをかける場合、割り込みをかけるデバイスは8ビットの割り込みベクトルをCPUに送る必要がある。

このベクトルはメモリ位置を指定するポインタの下位データとして使用され、指定された位置にはその割り込みデバイスに対するサービス・ルーチンのスタート・アドレスを与える。

デバイスからの8ビットのベクトルはポインタ（間接）の下位8ビットとなり、CPU内のIレジスタのデータが、そのポインタの上位8ビットとなる。

なお、PIOの各ポート（AとB）の割り込みのベクトルはプログラムによって指定するものであることに注意を要する。

ベクトルの最下位ビットはスタート・アドレスを連続する2バイト（16ビット）で指定しなければならないので、プログラマ側で0を指定しなければならない。

PIO自身でCPUデータ・バスに乗るRETI（割り込みからのリターン）命令を直接解続できるので、システム内の各PIOはCPUとの間に別の通信線を引かなくても、CPUの割り込みサービスが終了する時期を検知できるようになっている。

3. 端子説明

図3－1にZ－80A PIOの端子配置図を示す。以下この章では、各端子の機能を説明する。

D_0～D_7　　Z－80A CPUのデータ・バス（双方向性トライ・ステート）

このバスはCPU－PIO間ですべてのデータや命令などを転送するために使用される。

D_0はバスの最下位ビットである。

B/A SEL　　ポートB、Aの選択（入力、アクティブ〝H〟）

この端子で、CPU－PIO間でデータ転送を行う際に、アクセスされるポートを指定する。この端子への入力が〝L〟レベルのときポートAを選択し、〝H〟レベルのときポートBを選択する。

通常CPUからのアドレス・ビットA_1をポート選択用として使用する。

C/D SEL　　制御信号、データの選択（入力、アクティブ〝H〟）

この端子で、CPU－PIO間で行われるデータ転送の型を指定する。この端子への入力が〝H〟レベルのとき、PIOへの書き込みデータはB/A選択で指定されたポートへのコマンド（命令）として解釈され、この端子への入力が〝L〟レベルのときには通常のデータ転送となる。

通常CPUからのアドレス・ビットA_0を制御信号、データ選択用として使用する。

CE　　チップ・イネーブル（入力、アクティブ〝L〟）

この端子を〝L〟レベルにすると、PIOは書き込みサイクルの期間ではCPUからコマンドまたはデータを受けとることができ、読み出しサイクルの期間ではCPUへデータを送出することができる。この信号として、通常ポートA、ポートBの各データおよびコマンドのいずれかを指定できる入出力ポート・アドレスのデコード信号を使用する。

Φ　　システム・クロック（入力）

Z－80A PIO内部の同期用として、Z－80A CPUの標準システム・クロックを利用している。この信号は単相クロックである。

$\overline{M1}$　　CPUのマシン・サイクル1信号（入力、アクティブ〝L〟）

CPUからのこの信号はPIO内で制御用同期パルスとして使用される。

Z－80A CPUは、$\overline{M1}$がアクティブでかつ$\overline{RD}$がアクティブのときには、メモリからの命令をフェッチするが、一方$\overline{M1}$がアクティブで$\overline{IORQ}$がアクティブのときには、$\overline{INT}$入力に対する割り込みアクノリッジを出力していることを示している。

さらにPIOは$\overline{M1}$で次のような動作をする。

1．$\overline{M1}$にPIOの割り込み論理回路を同期させる。

2．$\overline{RD}$、$\overline{IORQ}$信号のいずれかがアクティブでないときに、$\overline{M1}$があれば、PIO内の論理回路はリセット状態になる。

$\overline{IORQ}$　　CPUの入力／出力要求（入力、アクティブ〝L〟）

$\overline{IORQ}$信号は、CPUとPIO間でコマンドやデータの転送を行う場合に、B/A選択、C/D選択、$\overline{CE}$、$\overline{RD}$信号などと組み合わせて使用される。

$\overline{CE}$、$\overline{RD}$、$\overline{IORQ}$がともにアクティブのとき、B/Aで選択されたポートからCPUへのデータ転送が行われる（読み込み動作）。

逆に、$\overline{CE}$、$\overline{IORQ}$がともにアクティブであり、$\overline{RD}$が非アクティブのとき、B/Aで選択されたポートへ、CPUからC/D選択信号で指定された情報（データまたはコマンド）が送られる。また、CPUが割り込み受け付け状態になれば、$\overline{M1}$と$\overline{IORQ}$がともにアクティブとなり、割り込みを出しているポート（優先順位が最上位で、かつ割り込み要求を出しているとする）から自動的にCPUデータ・バス上へポートの割り込みベクトルが送り込まれる。

$\overline{RD}$　　CPUの読み出しサイクル・ステータス（入力、アクティブ〝L〟）

$\overline{RD}$がアクティブのとき、メモリの読み出し、あるいは入出力デバイスの読み出しが行われる。

$\overline{RD}$信号は、B/A選択、C/D選択、$\overline{CE}$、$\overline{IORQ}$信号と組み合わせて、PIOからCPUへのデータ転送を行わせるのに使用される。

IEI 割り込みイネーブル入力（入力、アクティブ〝H″）

割り込みを発生するデバイスが2個以上ある場合、それらの割り込みの優先順位を決めるデージー・チェーンを形成するために用いられる。このPIOより優先順位の高いデバイスからのIEO信号が〝H″レベルであれば、条件が成立した場合に、このデバイス（PIO）から割り込みを発生することができる。（このPIOの直前のデバイスのIEOをこのPIOのIEIに接続しておく。）

IEO 割り込みイネーブル出力（出力、アクティブ〝H″）

このIEO信号もデージー・チェーンを形成するのに必要なものである。

IEIが〝H″レベルで、CPUがこのPIOからの割り込みのサービスを行っていないときにだけIEOは〝H″レベルとなる。このPIOの割り込みサービスが行われているときには、これより優先順位の低いデバイスのIEIがこの信号で〝L″レベルにおさえられ、そこからの割り込みは阻止される。（このIEOはこのPIOの直後のデバイスのIEIに接続しておく。）

$\overline{\text{INT}}$ 割り込み要求（出力、オープン・ドレイン、アクティブ〝L″）

PIOからのCPUに対する割り込み要求はこの信号をアクティブにして行う。この$\overline{\text{INT}}$（出力）はCPUの$\overline{\text{INT}}$（入力）に接続する。

A_0～A_7 ポートAバス（双方向性、トライ・ステート）

この8ビット・バスはZ−80A PIOのポートAと周辺装置との間でデータ、ステータス、制御情報の転送を行うもので、A_0はポートAデータ・バスの最下位ビットである。

$\overline{\text{A STB}}$ 周辺装置が与えるポートAストローブ・パルス（入力、アクティブ〝L″）

この信号の意味は、以下のように、指定されたモードごとに異なる。

1）出力モード：PIOから送り出されたデータを周辺装置が受け取ったことをPIOに対して通知する信号であり、この信号の立ち上がりが有効である。

2）入力モード：周辺装置からポートAの入力レジスタへデータをロードしたとき、その周辺装置から出されるストローブ信号として使用する。この信号がアクティブになったとき、データがPIOにロードされる。

3）双方向モード：この信号がアクティブのとき、ポートAの出力レジスタからのデータがポートAの双方向データ・バス上に乗せられる。
ストローブが立ち上がれば、周辺装置がデータを受け取ったとみなされる。

4）ビット制御モード：このときのストローブ入力は無効である。

A RDY レジスタAレディ（出力、アクティブ〝H″）

この信号の意味は、以下のように、指定されたモードによって異なる。

1）出力モード：ポートAの出力レジスタがロードされていて周辺用データ・バスが安定になり、周辺装置へのデータ転送が準備できたことを指示するため、アクティブとなる。

2）入力モード：ポートAの入力レジスタが空となり、周辺装置からのデータ受け入れ準備ができたとき、アクティブとなる。

3）双方向モード：ポートAの出力レジスタに周辺装置へ送るべきデータがロードされたとき、アクティブとなる。

4）ビット制御モード：この信号は強制的に〝L″状態になる。

B_0～B_7 ポートBバス（双方向性、トライ・ステート）

この8ビット・バスはデータ、ステータス、制御語などの情報をPIOのポートBと周辺装置間で転送するために用いられる。ポートBのデータ・バスは、ダーリントン・トランジスタを駆動できるよう、1.5V当たり1.5mAの出力能力をもっている。

B_0はバスの最下位ビットである。

$\overline{\text{B STB}}$ 周辺装置が与えるポートBストローブ・パルス（入力、アクティブ〝L″）

この信号の意味は、$\overline{\text{A STB}}$と同様であるが、次の点が異なる。ポートAの双方向モードでは、周辺装置からポートAの入力レジスタへのデータのストローブ（移し込み）をこのB STBで行う。（ポートAの出力レジスタ内容の出力ストローブは$\overline{\text{A STB}}$を使用する。）

B　RDY　　レジスタBレディ（出力、アクティブ"H"）

この信号の意味は、A　RDYと同様であるが、次の点が異なる。ポートAの双方向モードでは、ポートAの入力レジスタが空となり、データの受け入れ準備ができたとき、このB　RDYがアクティブとなる。（ポートAの出力レジスタ内容の出力準備完了の指示はA　RDYによる。）

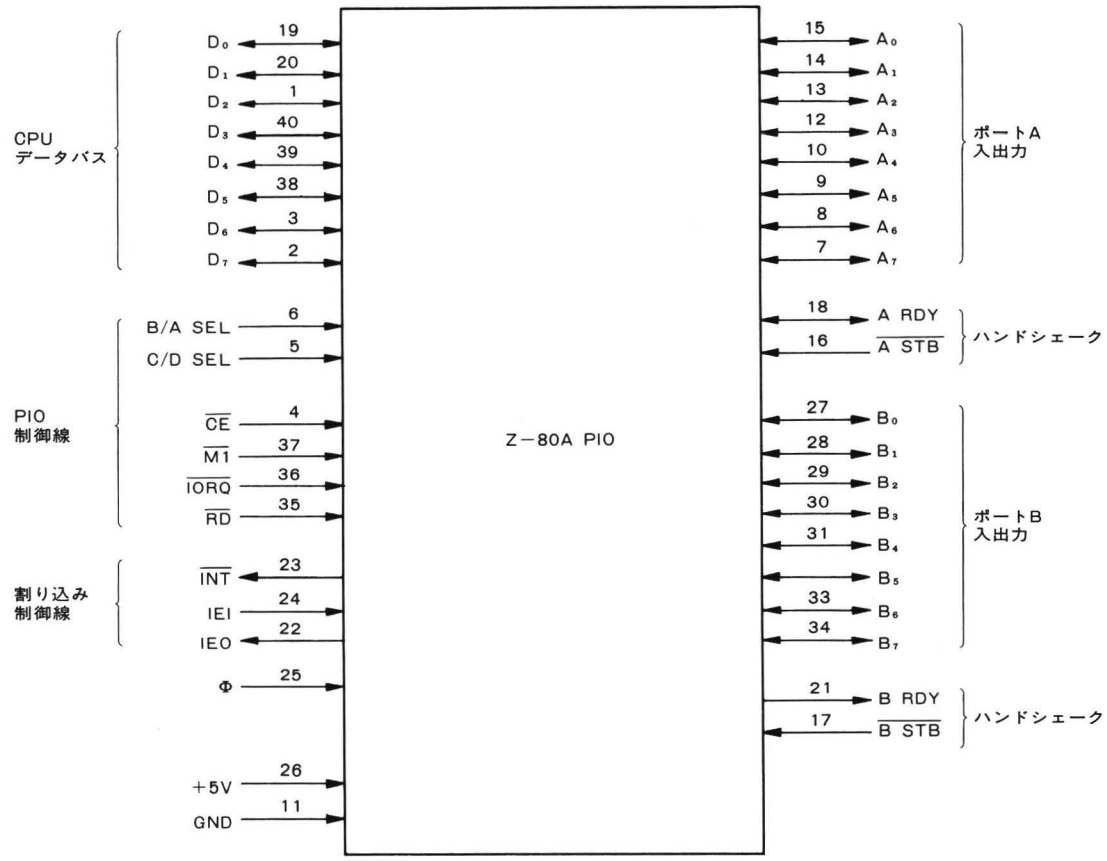



図3−1　　Z−80A PIO端子配置図

4. プログラミング

4.1 リセット

Z－80A PIOは、電源投入時、自動的にリセット状態に入る。リセット状態では次のような機能をもっている。

1）両ポートのマスク・レジスタがリセットされ、全データ・ビットが禁止（インヒビット）となる。

2）ポートのデータ・バス線は高インピーダンス状態になり、ハンドシェークのレディ信号は非アクティブ（"L"）となる。動作モードは自動的にモード1に設定される。

3）ベクトル・アドレス・レジスタはリセットされない。

4）両ポートの割り込みイネーブル・フリップ・フロップがリセットされる。

5）両ポートの出力レジスタもリセットされる。

電源投入時の自動リセットに加え、PIOのリセットは、RDあるいはIORQ信号を非アクティブにして2クロックのM1を与えても行える。

M1期間にRDかIORQかが検出されなければ、PIOはM1信号が非アクティブになった直後リセット状態になる。

このリセットの目的は、電源を切ってリセットするという手順を省き、簡単な外部ゲートのみでリセットを行うことができるようにするためであり、40ピンのパッケージに収める上での処置である。

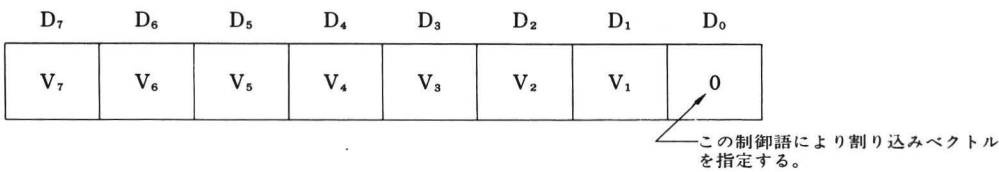
一度PIOが内部リセット状態に入ると、CPUから制御語を受けるまでそのままリセット状態を保持する。

4.2 割り込みベクトルのローディング

PIOは、モード2の割り込み応答のとき、Z－80A CPU側に情報を返して、CPUがこれを処理・操作できるように設計されている。

このモードでは、割り込みを要求しているデバイスから割り込みベクトルを供給することになるが、このベクトルはCPU内でそのデバイスの割り込みサービス・ルーチンのアドレスを形成するのに使用される。

このベクトルは、その時点で最高位の優先順位をもっているデバイス（PIOなど）から割り込みアクノリッジ・サイクル期間中に、Z－80Aデータ・バス上に乗せられる。(CPUの割り込みサービスの詳細については、第1部Z－80A CPUテクニカル・マニュアルを参照されたい。)

PIOが出す割り込みベクトルは次のようなフォーマットで、CPUから出力命令によって制御語を書き出し、希望するPIOにロードしておく。この場合のポート・アドレスはAまたはBポートの制御語に対応するアドレスである。

ベクトルの書き込みにおいてはD_0は、フラグ・ビットとして使用され、このビットが0である場合のV_1からV_7までが、PIO内のベクトル・レジスタにロードされる。

割り込みアクノリッジ期間に割り込みをかけたポートのベクトルが、Z－80Aデータ・バス上に、上のようなフォーマットで現われる。

4.3 動作モードの選択

PIOのポートAは次のような4種の異なったモードで使用できる。

モード0（出力モード）

モード1（入力モード）

モード2（双方向モード）

モード3（ビット制御モード）

数字は次のような関連で覚えるとよい。

0＝OUT, 1＝IN, 2＝BI-DIRECTIONAL

(出力)　(入力)　(双方向)

ポートBはモード2を除いたモード0、1、3で動作可能である。動作モードの指定はPIOに次のようなフォーマットの制御語を書き込んで行う。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
M1	M0	X	X	1	1	1	1

X＝使用しないビット
(0, 1いずれでもよい)

モード語

このデータ・フォーマットのとき
モード指定が行われる。

ビットD_7、D_6により、以下のようにモードを指定できる。

D_7	D_6	モード
0	0	0（出力）
0	1	1（入力）
1	0	2（双方向）
1	1	3（制御）

ビットD_5、D_4は無視される。（0、1いずれでもよい。）
ビットD_3～D_0により動作モードを指定し、1111に設定しておかねばならない。

モード0を指定した場合、ポートの出力レジスタ内のデータが対応するポートのデータ・バス上に乗せられる。この出力レジスタの内容は出力命令を用いて新しいデータにいつでも簡単に書き換えられるし、また、ある時点での出力レジスタの内容を入力命令を用いてCPU側へ呼び戻す（出力レジスタの内容は変化しない）こともできる。

モード0では、CPUから出力レジスタに対してデータの書き込みがあった場合、そのポートのハンドシェーク線であるレディ信号はアクティブ（〝H〟）となり、周辺装置にデータの転送が可能であるという指示を出す。この信号は周辺装置から受信したというストローブ信号が返されるまで、〝H〟レベルのままである。その返されたストローブの立ち上がりエッジで、もし割り込みが発生するようにプログラムされているならば、割り込みが発生し、レディ線は非アクティブ（〝L〟）となる。この簡単なハンドシェーク方式は多くの周辺装置によく使用されているものと同じ方式である。

モード1を指定した場合、ポートは入力モードとなる。ハンドシェーク動作を始めるためには、まずそのポートに対して読み出しを行うだけでよい（ダミー・リード）。この動作によってレディ線がアクティブとなり、〝周辺装置へ入力レジスタが空になったのでデータをロードせよ〟という指示が出される。次に周辺装置はストローブ信号に同期してデータを送出するが、PIOはこのストローブ信号がアクティブになることによって内部入力レジスタにデータを取り込む。この返されたストローブの立ち上がりエッジで可能であれば割り込みが発生し、レディ信号が非アクティブ状態に落される（〝L〟となる）。データがオーバラン状態にならないように注意が行き届いていれば、レディ信号の状態にかかわらず、ストローブをかけてデータを入力レジスタへロードしてもよい。
モード2の双方向データ転送はすべてのハンドシェーク線（A、B各2本）を使用して行う。したがって、モード2では、ポートAだけをデータの入出力用として使用し、ハンドシェーク線はポートAの2本を出力制御用とし、ポートBの2本を入力制御用として使用する。A RDY、B RDYは両方とも、同時にアクティブとなる。モード0とモード2における出力時の動作の違いは、このモード2では双方向にデータ転送を行うのを可能にするためA STBがアクティブになったときにだけポートAの出力レジスタ上のデータがポートAのデータ・バス上に乗せられる、という点のみである。

モード3の動作はステータスを調べるためと制御用として応用するためのものであり、この場合ハンドシェーク信号は使用しない。モード3を選択すると次にポート・アドレスに書き込む制御語によって、ポートのデータ・バス線のどれを入力用とし、どれを出力用とするかを指定する必要がある。あるビットを1に設定すれば、そのビットに対応するデータ・バス端子は入力用となり、0に設定したビットに対応するデータ・バス端子は出力用となる。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
I/O_7	I/O_6	I/O_5	I/O_4	I/O_3	I/O_2	I/O_1	I/O_0

モード3の動作時、ストローブ信号は無視され、レディ線は〝L〟レベルに保持される。ポートとCPU間のデータのやりとりは動作中任意の時点で行える。ポートを読む場合、CPUに入るデータは入力として指定されたポートのデータ・バス線からのデータと出力として指定されたポートのデータ・バスからのデータが重なり合っている。

4.4　割り込み制御語の設定

各ポートへの割り込み制御語の設定は次のようなフォーマットで行う。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
割り込み イネーブル	AND/ OR	High/ Low	マスク 指定	0	1	1	1
	モード3のみで使用			割り込み制御語を指定する。			

ビット$D_7 = 1$のとき、対応するポートの割り込みイネーブル・フリップ・フロップがセットされ、ポートからの割り込み要求が可能となる。$D_7 = 0$のとき、イネーブル・フラグはリセットされ、割り込みを発生できない。イネーブル・フラグがセットされていて、割り込みが保留状態ならば、CPUの割り込み要求信号であるINT線は〝L″レベルのままである。ビットD_6、D_5、D_4はモード3でのみ使用される。割り込み制御語のビットD_4がセットされると、どのモードで動作している場合でも、未処理の割り込みはすべてリセットされる。モード3で、入出力線群のどれかがある指定状態になったとき、割り込み動作を実行できるようにするための条件設定にこの3ビットを用いる。ビットD_6でモード3のデータ一致条件をANDで採るか、ORで採るかを定め、$D_6 = 1$ならばAND、$D_6 = 0$ならばORとなる。たとえば、ANDの場合、入力線の指定ビットがすべてアクティブのときにのみ、割り込み発生の条件となり、ORの場合、指定ビットのいずれかがアクティブならば、割り込み発生の条件となる。

ビットD_5は入力線のアクティブ状態の極性規定するためのものである。$D_5 = 1$ならば、入力線が〝H″レベルのときアクティブとみなし、$D_5 = 0$ならば、入力線が〝L″レベルのときアクティブとみなす。

ビット$D_4 = 1$のとき、同一ポートに次に書き込まれた制御語はマスクと解釈される。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
MB_7	MB_6	MB_5	MB_4	MB_3	MB_2	MB_1	MB_0

このマスクでビット0の箇所の入力線のみを調べ、設定条件と一致すれば割り込みを発生させる条件となる。

5. タイミング

5.1 出力モード（モード0）

図5－1にモード0の動作のタイミングを示す。出力サイクルはつねにCPUの出力命令の実行でスタートする。

CPUから入出力への書き込み動作時にPIO内でWR*パルスが発生し（これは外部には出ない)、この信号によってCPUデータ・バス上のデータを指定されたポート（AかB）の出力レジスタにとり込む。

WR*パルスが立ち上がったあとの最初のΦの立ち下がりエッジで、周辺装置へのデータ供給が可能であるという指示を出すために、レディ・フラグがセットされる。たいていのシステムではこのレディ信号を周辺装置内のデータ・ラッチ信号として使用することができる。(WR*は図参照)

レディ信号は次のいずれかの状態になるまでアクティブのままである。

(1) 周辺装置がデータを受けとったときに返してくるストローブ信号の立ち上がりエッジで非アクティブとなる。

(2) すでにアクティブであって、さらに続いてポートの出力レジスタに、データを書き込む場合、IORQの立ち下がりエッジの3/2Φ後にレディ信号を強制的に非アクティブ（〝L″）にする。次にIORQの立ち上がりエッジのあとの最初のΦの立ち下がりエッジで、〝H″レベルに戻る。

このことからもわかるように、ポートのデータが変化すればレディ信号は必ず〝L″レベルとなるようになっている。レディ信号は、クロックΦの立ち下がりエッジまでは、非アクティブにならない。クロックが立ち下がるまでレディ信号の立ち下がりを遅らせた理由はこれによって非常に簡単にストローブ・パルスを作ることができるようになるからである。

レディ線とストローブ線とを直接結べば、他の回路を使用しなくても1クロック期間のストローブを作ることができる。IFFがセットされていて、かつ割り込みを要求しているデバイスがその時点で最高位の優先順位をもっていれば、ストローブ・パルスの立ち上がりエッジで自動的にINTが発生する。

もし、PIOがリセット状態になければ、モード0を指定する前に出力レジスタに情報をロードすると、モード0を指定したときに、すでにロードしている出力レジスタの情報がそのポートの出力線から出力する。

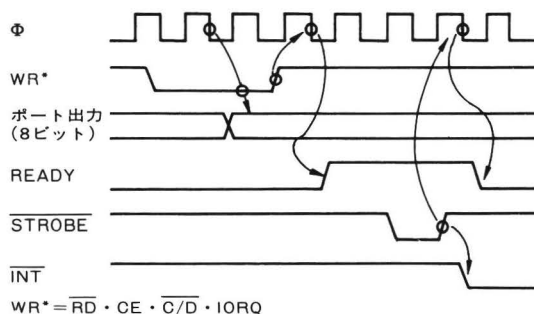



図5－1　モード0（出力）のタイミング

5.2 入力モード（モード1）

図5－2に入力サイクルのタイミングを示す。

CPUがデータの読み込みを完了したのち、周辺装置はストローブ線を使用してこのサイクルを始める。このストローブ信号が〝L″レベルになることにより、ポートの入力レジスタへデータをロードし、条件が満たされていれば、ストローブ信号の立ち上がりエッジで割り込み要求線（INT）がアクティブとなる。入力レジスタが満杯になれば、クロック（Φ）の次の立ち下がりエッジで、レディをリセット（非アクティブ）し、CPUがデータを読み込んでしまうまで、次のデータのロードを禁止する。

そこでCPUは割り込みサービス・ルーチンに入り、そのルーチンで割り込みをかけたポートからのデータを読み込む。読み込み終了後、CPUのRD信号によるRD*（図5－2参照）の立ち上がりエッジのあとのΦの立ち下がりのときにレディが立ち上がり、次の新しいデータをPIOへロードしてもよい状態になる。

すでにリード待ち状態（アクティブ）であって、データがPIOポートからCPUに読み込まれている間、IORQの立ち下がりエッジのあとの3/2Φ後にレディは強制的に〝L″レベルに落ちる。

ユーザはレディが〝H″のときにのみPIOにデータを送り込むようにすれば、CPUがPIOからデータを読み出す期間はレディが〝L″レベルとなっているので、読み出し途中でPIOの入力レジスタの内容が変ってしまうことはない。

レディは、すでに述べたように、IORQの立ち上がりエッジのあとで再び〝H″レベルになる。

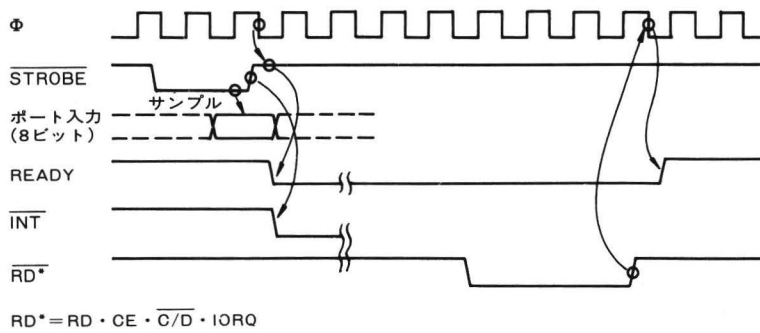



図5－2　モード1（入力）のタイミング

5.3 双方向モード（モード2）

このモードはたんにモード0とモード1の組み合わせであり、4本のハンドシェーク線をすべて使用する方式である。

このモードでは4本のハンドシェーク線が必要なので、データ転送にはポートBは使用せずポートAのみを使用する。このとき、ポートBはビット制御モードにセットしておく必要がある。ポートBをモード3にした場合の割り込みと、ポートAをモード2で動作させかつその入力転送中に生じる割り込みのベクトルは同じものになる。これら2つのモードのベクトルの混同はポートBを制御モード（モード3）で動作させ、かつマスクレジスタの全ビットをインヒビットすることによって回避できる。

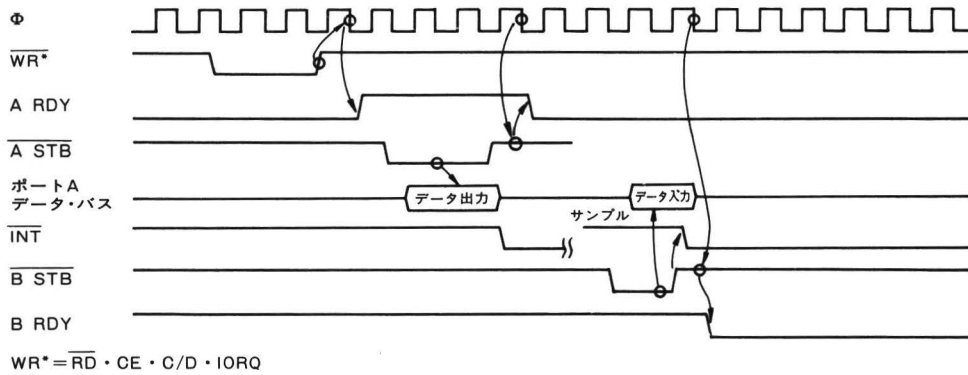



図5－3　モード2（双方向）のタイミング、ポートA

双方向モードでのタイミングを図5－3に示す。これはほとんどモード0、モード1で述べたのと同様であり、ポートAのハンドシェーク線を出力制御用に使用し、ポートBの線を入力制御用として動作させると考えればよい。その場合の2つのモードの異なる点は、モード2ではAストローブが〝L〟レベル（A　STBがアクティブ）になったときにのみ、データがバス上に乗せられることである。

このストローブの立ち上がりエッジのあとしばらくデータは安定しているので、この立ち上がりエッジを周辺装置へのラッチ信号として使用することができる。

モード2の入力動作はモード1のそれと同じである。

ポートA、ポートBともに割り込み駆動で双方向転送が行えるようにするためには、それぞれが割り込み機能をもつようにしておく必要がある。

A　STBがアクティブのときには、周辺装置からポート・データ・バス上へデータを乗せてはならない。ちなみに周辺装置からのデータをB　STBでゲートしておけば、バス駆動の競合状態は生じない。PIOはこのデータをB　STBの〝L〟レベルで取り込むようになっている。

PIOは、このモードでデータの取り込みを行う際に、取り込みに必要なホールド時間が零であってもよいように設計されているので、上述のような簡単なゲート方式でよい。すなわち上述のストローブ（B　STB）の立ち上がりエッジの直後で、バスからのデータは取り込み禁止（ディセーブル）となる。

5.4 ビット制御モード（モード3）

ビット制御モードでは、ハンドシェーク信号は使用しないので、任意の時点で正規のポートの書き込み、読み出しが行える。

書き込み時、データはモード0と同じタイミングで出力レジスタにラッチされる。もしポートAがモード3で動作中には、A RDYは〝L〟レベルに固定される。ポートAをモード2で使用していなければ、ポートBがモード3で動作する場合はいつもB RDYは〝L〟レベルに固定される。この場合、B RDYの状態は何によっても変化しない。

PIOからの読み出しで、CPUへ返るデータは、出力として指定しているポート・データ線からの（出力レジスタの）データと入力として指定している同じポートのデータ線からの（入力レジスタの）データとの合成になっている。入力レジスタの内容はRDが下がる直前にポート・データ・バス上にあったデータに等しい。（図5－4参照）

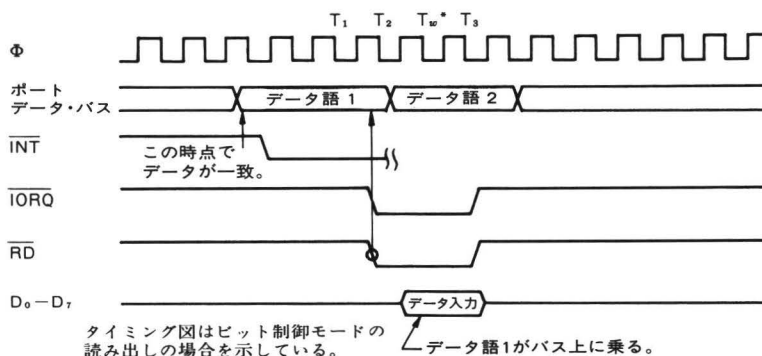



図5－4　ビット制御モードのタイミング

ポートからの割り込みが受け付け可能で、ポート・データ線上の入力データが8ビット・マスクと2ビットのマスク制御レジスタで指定される条件に一致すれば（論理式が等しくなる）、割り込みが発生し、条件の一致状態に変化がなければ、新しく割り込みが発生することはない。

モード3の割り込みは、このモードでの論理演算が〝偽〟から〝真〟に変ったときにのみ、発生する。たとえば、モード3での論理式が〝OR〟機能であれば、マスクされないポート・データ線の少なくともいずれかひとつがアクティブになれば割り込み要求が出る。

次に先の状態に続いて他のマスクされないデータ線がアクティブになっても、モード3の論理演算の結果に変化はないので、新たに割り込み要求が出ることはない。

論理演算の結果がM1サイクルの直前かその期間中に〝真〟になれば、割り込み要求はM1の立ち上がりエッジで発生する。

6. 割り込みサービス

PIOから割り込み要求を受け取ったあと、CPUは割り込みアクノリッジを送出する。(このアクノリッジ信号としてM1とIORQが同時に出る。)

この期間において、PIO内の割り込み論理回路は割り込み要求を出している最も優先順位の高いポートを決定する。(これは単にIEIが〝H″レベルでIEOが〝L″レベルのデバイスを求めることで決められる。)

デージー・チェーンのイネーブル線の状態を確実にするため、M1がアクティブになったとき、各デバイスの割り込み要求状態の変更は許されない。

割り込みアクノリッジの期間に、割り込みを出している最も優先順位の高いデバイスがCPUデータ・バス上にその割り込みベクトルを乗せる。

図6－1に割り込み要求に関係したタイミングを示す。M1期間では、新しい割り込み要求を発生させることができない。これは4個までのPIOからの割り込みイネーブル信号を受けるに十分な時間をとる必要があるからである。

INTAの期間で、IEIが〝H″レベル、IEOが〝L″レベルであるPIOがそのポートの8ビットの割り込みベクトルをデータ・バス上に送り込むことになる。

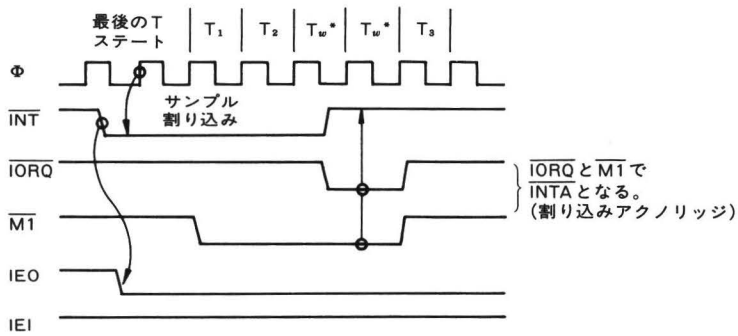



図6－1　割り込みアクノリッジのタイミング

PIOからの割り込み要求が、受け付けられると、要求を出したポートが〝サービス″を受ける。IEIが〝H″レベルであれば、このポートのIEOはRETI命令を実行すると〝L″レベルから〝H″レベルに変化する。

割り込み要求が受け付けられていないPIOはOPコード〝ED″(入出力命令などのOPコード)を解読後、次のM1の1サイクル間IEOを強制的に〝H″レベルにする。

この動作により、この2バイトのRETI命令を選ばれたPIOポートだけが解読することができるようにしている。(図6－2参照)

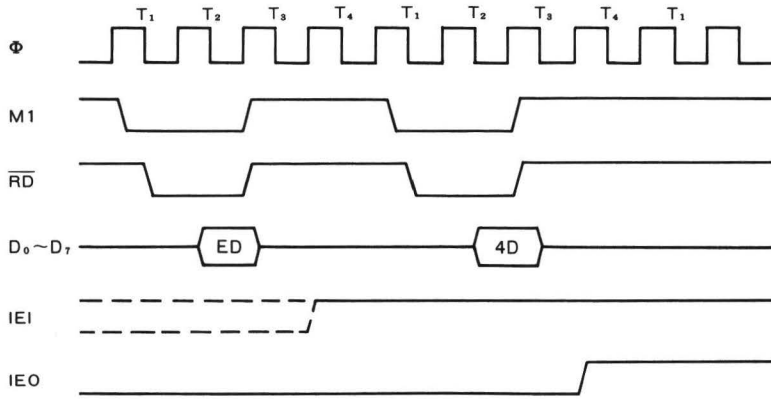



図6－2　割り込みサイクルからの復帰

4つのポートをデージー・チェーンで接続してある場合のネスト構造の割り込みシーケンスの例を図6－3に示す。

まず、ポート2Aが要求を出し割り込みが受け付けられるが、このポートがサービスされている途中で、優先順位の高いポート1Bが要求を出すと、これが受け付けられ、ポート2Aのサービスは保留される。

優先順位の高いポート1Bに対するサービス・ルーチンが完了し、RETI命令を実行して、そのルーチンの終了をポート2Aに知らせる。次に優先順位の低いポート2Aのサービス・ルーチンを再開する。

1．割り込み発生前のデージー・チェーンの状態

2．ポート2Aが割り込みを要求し、受け付けられる。

3．ポート1Bが割り込みを発生し、2Aのサービスを保留する。

4．ポート1Bのサービスを完了し、"RETI"を実行する。ポート2Aのサービスを再開する。

5．ポート2Aのサービス完了後2番目の"RETI"を実行する。

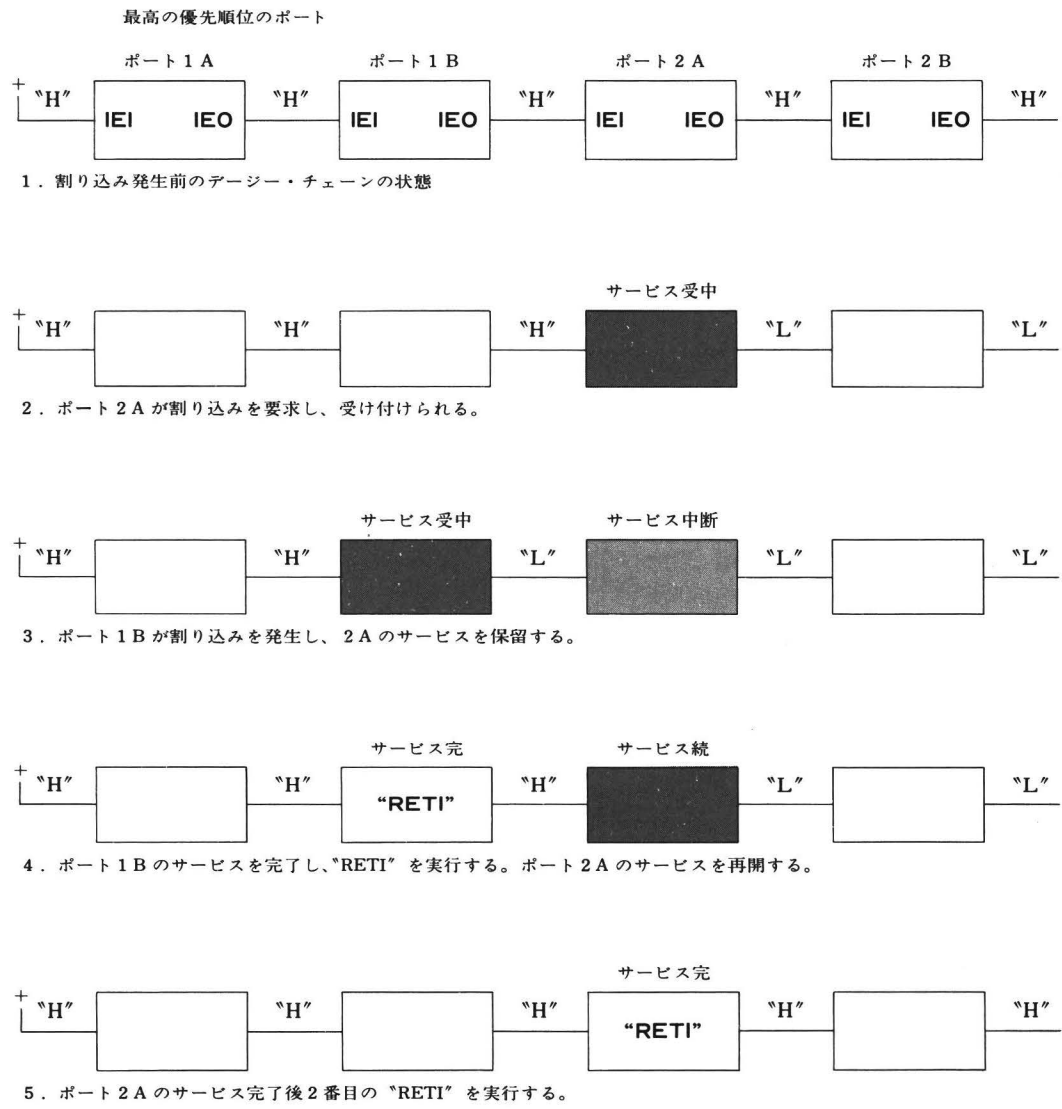



図6−3　デージー・チェーンでの割り込みサービス

7. 応用例

7.1 割り込みデージー・チェーンの拡張

割り込みの優先順位を付けるためのデージー・チェーンには、Z－80A PIO を外部回路を付加しないで最大4個まで接続できる。

この数の制限は、割り込みイネーブル信号（IEO）が各PIO を伝播していくときの遅延時間によるもので、この遅れは割り込みアクノリッジ・サイクルにおけるM1の立ち下がりエッジとIORQ の立ち下がりまでの間の時間からはみ出してはいけない。

割り込みイネーブルの状態をM1期間では変化させることができない。これによりCPU に戻るPIO からのベクトル・アドレスは、割り込みを要求した最高位の優先順位をもつデバイスからのものであると保証できる。

PIO を4個以上接続する必要がある場合は、図7－1に示すような〝ルック・アヘッド〟構成にするとよい。この方法で標準TTL を使用して30個以上の PIO を接続できる。

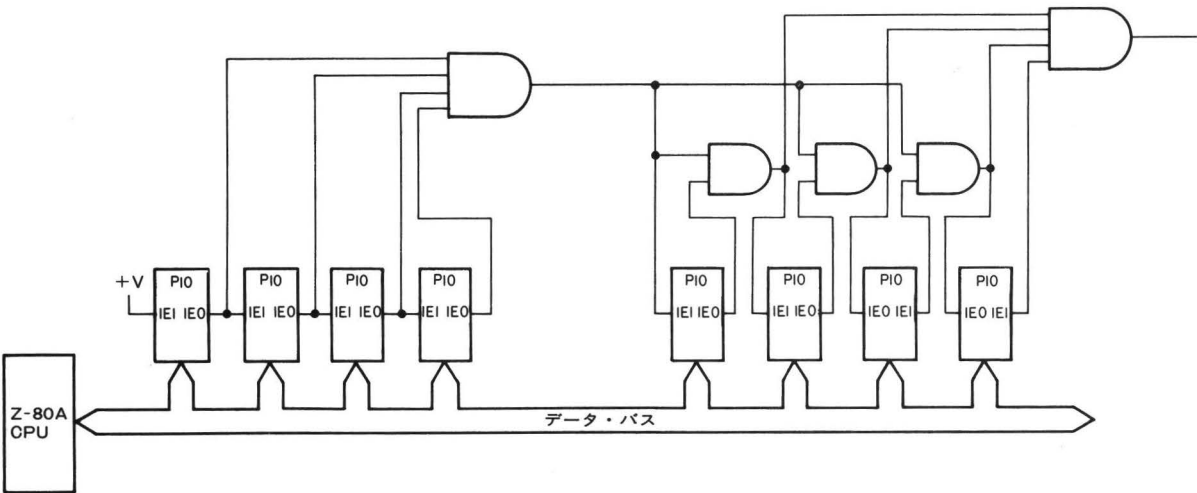



図7－1　デージー・チェーンの拡張方法

7.2 入出力インターフェース

Z－80A PIO を入出力ターミナル・デバイスと結合して、図7－2に示すような8ビット並列の双方向データ・バスを介してデータの送受を行う場合、ポートAに次のような制御語を送って、PIO をモード2（双方向モード）に設定すればよい。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	X	X	1	1	1	1

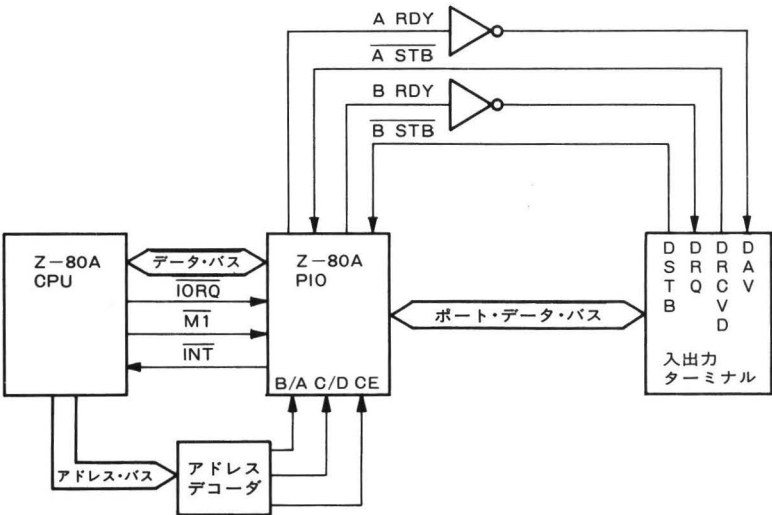



図7－2　入出力インターフェースの例

次に、適当な割り込みベクトルをロードする。(割り込みの操作については、　A.1　Z－80A CPU テクニカルデータを参照のこと。)

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
特殊テスト	パワーオン	電源異常アラーム	プロセス停止	温度アラーム	ヒータオン	圧力系	圧力アラーム

M1が割り込みアクノリッジ・サイクル時のものでなければ、割り込みモード語をセットしたあとの最初のM1の立ち上がりエッジで割り込みがイネーブル（受け付け可能）となる。

割り込みモード語の後にマスク語が続いて送られたときは、マスクがセットされたあとの最初のM1の立ち上がりエッジで割り込みはイネーブルとなる。

こうして、周辺装置とCPUとの間でデータの転送が可能となる。この転送時のタイミングは第5章で述べたことと同様である。

7.3　制御用インターフェース

ビット制御モードでの例を図7－3に示す。

工業プロセスでのモニタを考えてみる。異常動作状態の発生をZ－80A CPUを基本とした制御システムに伝達する場合、次のようにフォーマットのプロセス制御語およびステータス語を設定しておくとする。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	0

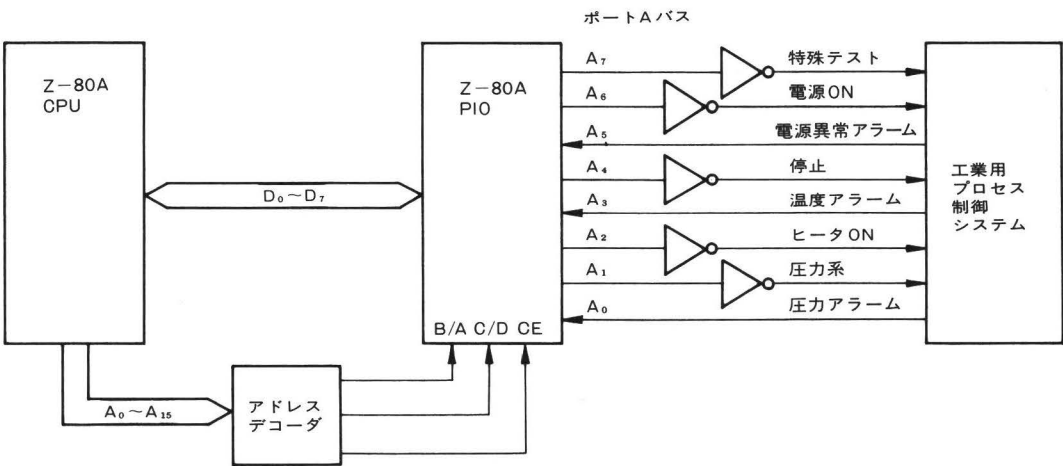



図7－3　　制御モードの応用

PIOを次のように使用する。ポートAに次のような制御語を書き込んでモード3に設定する。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	1	X	X	1	1	1	1

モード3を指定した場合、その次の制御語は入出力選択語でなければならないので、たとえば、ポート・データ線 A_5、A_3、A_0 を入力とする場合には次のような語を書き込む。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	1	0	1	0	0	1

次に必要な割り込みベクトルをロードしておく。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	0

割り込み制御語を続いてポートに転送する。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	1	1	0	1	1	1
割り込み イネーブル	OR 論理	アクティブ "H"	マスク	割り込み制御			

割り込みモードを指定したのち、次に書き込むマスク語を以下のように設定する。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	1	0	1	0	1	1	0

モニタ線の指定（D_5、D_3、D_0）

以上の準備で、A_5、A_3、A_0 いずれかの線の検出器からの信号が "H" レベルになれば、割り込み要求が発生する。

マスク語により入力、出力のどのような組み合わせに対しても割り込みを発生させる条件を選択、指定できる。たとえば、上述のマスク語が次のようであれば、

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	1	0	1	0	1	1	0

割り込み要求は出力レジスタのビット A_7（特殊テスト）をセットしたときにも発生することになる。

ポートの指定方法については、たとえば次のようにするとよい。

$E0_H$＝ポートA　データ
$E1_H$＝ポートB　データ
$E2_H$＝ポートA　制御用
$E3_H$＝ポートB　制御用
（H は前に置かれた数が16進表現であることを示す。）

ポート番号をこのように指定しておくと、アドレス・バスの A_0 をポートB/Aの選択用、アドレス・バスの A_1 をC/D（制御信号/データ）の選択用として使用することができ、便利である。

チップ選択（CE）には、A_2 から A_7 までのCPUのアドレス・ビットをデコードすればよいが、この場合、A_7～A_2 が11000になればチップ選択される。

周辺装置が少ない場合には、CEとして特別にデコードする必要はなく、必要に応じて A_2 から A_7 の適当なビットを直接使用することもできることに注意されたい。

8. プログラミングのまとめ

8.1 割り込みベクトルのロード

V_7	V_6	V_5	V_4	V_3	V_2	V_1	0

8.2 モードの設定

M1	M0	X	X	1	1	1	1

M1	M0	モード
0	0	出　力（0）
0	1	入　力（1）
1	0	双方向（2）
1	1	ビット制御（3）

モード3を指定した場合、次に書き込むデータにより入出力レジスタを設定する。

I/O_7	I/O_6	I/O_5	I/O_4	I/O_3	I/O_2	I/O_1	I/O_0

I/O＝1ならば、入力ビットとなる。
I/O＝0ならば、出力ビットとなる。

8.3 割り込み制御語の設定

割り込みイネーブル	AND/OR	High/Low	マスク指定	0	1	1	1

（AND/OR、High/Low、マスク指定：モード3でのみ使用）

マスク指定（Mask follows）が〝H〟レベルならば、次にPIOに書き込むデータはマスクとなる。

MB_7	MB_6	MB_5	MB_4	MB_3	MB_2	MB_1	MB_0

MB＝0ならば、モニタ・ビットとなる。
MB＝1ならば、非モニタ・ビットとなる。

また、ポートの割り込みイネーブル・フリップ・フロップ（IFF）を次のようなコマンドを使用して、割り込み制御語の他の部分を変えずにセットまたはリセットすることもできる。

割り込みイネーブル	X	X	X	0	0	1	1

A.3 MZ-2000の仕様

1. MZ-2000一般仕様

項目	仕様
CPU	SHARP LH0080A(Z80A-CPU)
クロック	4MHz
メモリ	ROM 2Kバイト (IPL:イニシャル・プログラム・ローダ) ROM 2Kバイト (キャラクタ・ジェネレータ) RAM 64Kバイト (ダイナミックRAM)
CRT ディスプレイ	10型CRT (グリーンフェイス) キャラクタ表示 8×8ドットマドリックス 1) 1,000キャラクタ (40キャラクタ×25行) 2) 2,000キャラクタ (80キャラクタ×25行) 1)、2):ソフトで切り替え グラフィック表示 (オプション) 1) 320×200ドット(標準BASIC) 2) 640×200ドット(標準BASIC) 3) カラーコントロール(拡張BASIC) ページ1、2、3およびキャラクタとの混合表示が可能 1)、2):ソフトで切り替え
カセットテープ・デッキ	標準オーディオカセットテープ使用 データ転送速度 2,000ビット/秒 データ転送方式 SHARP PWM方式 マニュアルおよびソフトウェアによりコントロール
サウンド出力	最大 400mW(440Hz)
キー構成	キー数 92 ASCII標準、10数字キー ファンクションキー、カーソルコントロールキー カセットテープデッキコントロールキー
編集機能	カーソルコントロール:上、下、左、右、 ホーム、クリアキー 挿入、削除キー
時計機能	内蔵
電源	AC100V±10% 50/60Hz 消費電力 約50W
温度	使用温度 0℃~35℃ 保存温度 -15℃~60℃
湿度	使用湿度 80%以下
重量	約 13kg
外形寸法	幅 440mm 奥行 490mm 高さ 262mm

2. CPUボード部仕様

項目	仕様
CPU	SHARP LH0080A(Z80A-CPU) 1個
PIO	SHARP LH0081A(Z80A-PIO) 1個
ROM	IPL ROM (2Kバイト) 1個 キャラクタジェネレータROM (2Kバイト) 1個
RAM	標準装備 64KビットRAM 8個 (64Kバイト) ビデオ RAM 2KバイトRAM 1個
CRT コントローラ	SHARP LZ90D01 1個
メモリーコントローラ	SHARP LZ90D02 1個
プログラマブルカウンタ	8253 1個
プログラマブル周辺インターフェイス	8255 1個
その他のIC	18個

3.カラーグラフィックボード部仕様 (オプション……ページ1含む)

項目	仕様
カラーグラフィックコントローラ	SHARP LZ90D03 1個
RAM	16KビットRAM 8個(16Kバイト)* 最大48Kバイト
その他のIC	21個

*ページ2、3はオプション (各8個…16Kバイト)

4. 電源部仕様

項目	仕様
入力	AC100V±10% 50/60Hz
出力	5V、-5V 12V、15V

5. ディスプレイ部仕様

項目	仕様
サイズ	10型
水平垂直周波数	垂直　60Hz　　水平　15.75KHz
電源	DC12V　　1.1A±10%
ピクチャーチューブ	E2728B31　10型　90°偏向防爆型 ヒーター　12V　75mA
IC	2個
トランジスタ	7個
ダイオード	13個
音声出力	最大400mW　(440Hz) スピーカ　8cm ラウンドダイナミックタイプ32Ω
コントロールツマミ	Volume　V-Hold Contrast　H-Hold Brightness　Focus
動作温度	−10℃～50℃

項目	仕様
ビデオアウトプット	標準40Vpp　　(LIMIT35Vpp)
解像度	水平　画面中央部で左の模様が確認できること
非直線性歪	水平　+8%　(±14%MAX) 垂直　±8%　(±12%MAX)
幾何学的歪	糸巻歪　1%　(2%MAX) 樽形歪　1%　(2%MAX) 台形歪　1%　(2%MAX) 平行四辺形歪　1°　(2.5°MAX)
高圧	0 beam　11.0KV　(10.0KVMIN) (12.0KVMAX)
電源	DC　12.0V　1.05A　(1.2AMAX)
電源動作範囲	12V±10%
スキャンサイズ	水平　10%　(15%MAX) 垂直　10%　(15%MAX)
水平引込範囲	±300Hz　(LIMIT±100Hz)
垂直引込範囲	−12Hz　(LIMIT−6Hz)
音声周波数特性	440Hz(0dB) −10dB±4dB at 100Hz −12dB±4dB at 10KHz

6. カセット部仕様

項目	仕様
システム	PWM録音方式
定格電源	5V±5% 12V±5%(安定化) 15V(非安定化)
使用半導体	トランジスタ　2個 IC　12個 ダイオード　10個
使用テープ	C15～C60使用可
テープ定格速度	4.75cm/sec

項目	仕様
トラックの数及び用法	2トラックモノラル方式
モータ	12V電子ガバナモーター
バイアス方式	DC
消去方式	DC
標準再生点	667μsec～333μsec
使用温度範囲	−10℃～+40℃
保存温度範囲	−25℃～+65℃

〔注意〕製品定格および仕様は予告なく変更することがあります。その場合ここに掲げた項目と異なる場合がありますから注意が必要です。

A.4 取り扱い上の注意

■電源コードについて

電源コードを机やイスの下に敷いたり、ものにはさんで傷をつけないようご注意ください。電源コードに傷がついたまま使用すると危険です。また電源コードを抜くときは必ずプラグを持って抜いてください。

■電源電圧について

電源電圧はAC100Vでご使用ください。電源電圧が極端に高かったり、低かったりすると故障の原因になり、十分性能が発揮できない場合があります。このようなときはお買い求めの販売店あるいはもよりのシャープお客様ご相談窓口、シャープエンジニアリング・サービスステーションにご相談ください。

■風通しについて

本機は温度上昇を防ぐため、キャビネットに通風孔があけてあります。風通しの悪い狭い場所に押し込んだり、布を掛けたり、カーペットやフトンの上に置いたりして、通風孔をふさがないでください。

■湿気やほこりについて

本機を湿気の多い場所や、ほこりの多い場所に置かないでください。故障の原因になります。

■高温について

本機を直射日光の当る場所や暖房器具のような熱器具の近くに置かないでください。キャビネットや内部の部品をいためる原因になります。

■水や異物について

本機の内部に水や液状のもの、針やピン等金属類が入ったまま使用すると危険です。異物が入らないようご注意ください。とくに水や液状の異物が入った場合すぐに電源コードのプラグを抜き、ご販売店あるいはもよりのシャープエンジニアリング・サービスステーションにご連絡ください。

■衝撃について

本機は精密な電子部品でできています。落としたり、物を当てたりして衝撃を与えないでください。故障の原因になります。

■異常のときは

万一故障した時や異常を感じたら使用を中止し、お買い求めの販売店あるいはもよりのシャープエンジニアリング・サービスステーションにご相談ください。

■長期間使用しない場合

長い間ご使用にならない場合は必ず電源コードのプラグをコンセントから抜いてください。

■周辺機器の増設

シャープ指定の部品または機器をご使用ください。指定部品または機器以外の使用ならびに改造は故障の原因になる恐れがあります。

■汚れ

本機の汚れはやわらかい布に水または洗剤を含ませて軽くふいてください。ベンジン、シンナーなど揮発性のものは使用しないでください。ケースの変色などの原因になります。

■雑音対策

雑音の多い環境では電源に混入する雑音をラインフィルター等で除去してください（ラインフィルターについてはもよりのサービスステーションでご相談ください)。信号ケーブルと他の機器や電源コードとは、できるだけ離して配置してください。

■雑音電波

ラジオやテレビなどの近くでご使用になりますとラジオやテレビに雑音が入ることがあります。なお、テレビなどの強い磁界を発生するものから影響をうけることがありますので、２～３ｍ以上離れた所でご使用ください。

■電源の「入り」・「切り」

電源スイッチの入り・切りは10秒以上の間隔をあけて操作してください。マイクロコンピュータの動作を確実にするため必要です。またスイッチの入った状態で電源プラグを差し込むと、故障の原因になります。

■カセットの取り扱い

カセット内の録音・再生ヘッドが汚れますと正確なデータの録音・再生ができません。１ヶ月に一度掃除をしてください。一般市販のクリーニングテープを使用すると便利です。

■テレビ画面のヤケについて

本機のテレビ画面は、長時間連続して同じ点を表示し続けると、その箇所にヤケを生じることがありますので、注意が必要です（やむを得ず長時間ご使用になる時は、後側のブライト調節ボリュームを充分しぼってご使用ください)。

■本体上部キャビネットの開き方

本機の上部キャビネットを持ち上げたら支持アームで上部キャビネットを支えます。本機の上部キャビネットを開いた状態での使用及び保管をしないでください。

シャープ株式会社

本　　　　社 〒545　大阪市阿倍野区長池町22番22号　電話 大　阪(06) 621-1221(大代表)
産業機器事業本部 〒639-11 奈良県大和郡山市美濃庄町492番地　電話 大和郡山(07435)3-5521(大代表)

●お客様ご相談窓口およびシャープエンジニアリング

北海道 (011)642-4649　東　北 (0222)96-4649　関　越 (0286)35-1151
東　京 (03)893-4649　北　陸 (0762)49-4649　中　部 (052)322-4649
近　畿 (06)643-4649　中　国 (082)874-4649　四　国 (0878)33-4649
九　州 (092)572-4649　沖　縄 (0988)62-2231